Introduction to Recurrent Neural Network

Prof. Chia-Yu Lin

Yuan Ze University

2021 Spring

Thanks to the slides from Prof. Hung-Yi Lee in NTU.

Outline

- Industry 4.0 problem
- Why RNN?
- RNN Model
  - Recurrent Neural Network (RNN)
  - Variants of RNN
  - Long Short-term Memory (LSTM)
- Learning in RNN
- More applications of RNN
- RNN for anomaly detection
- Summary

Why RNN

Can machine understand the meaning of words?

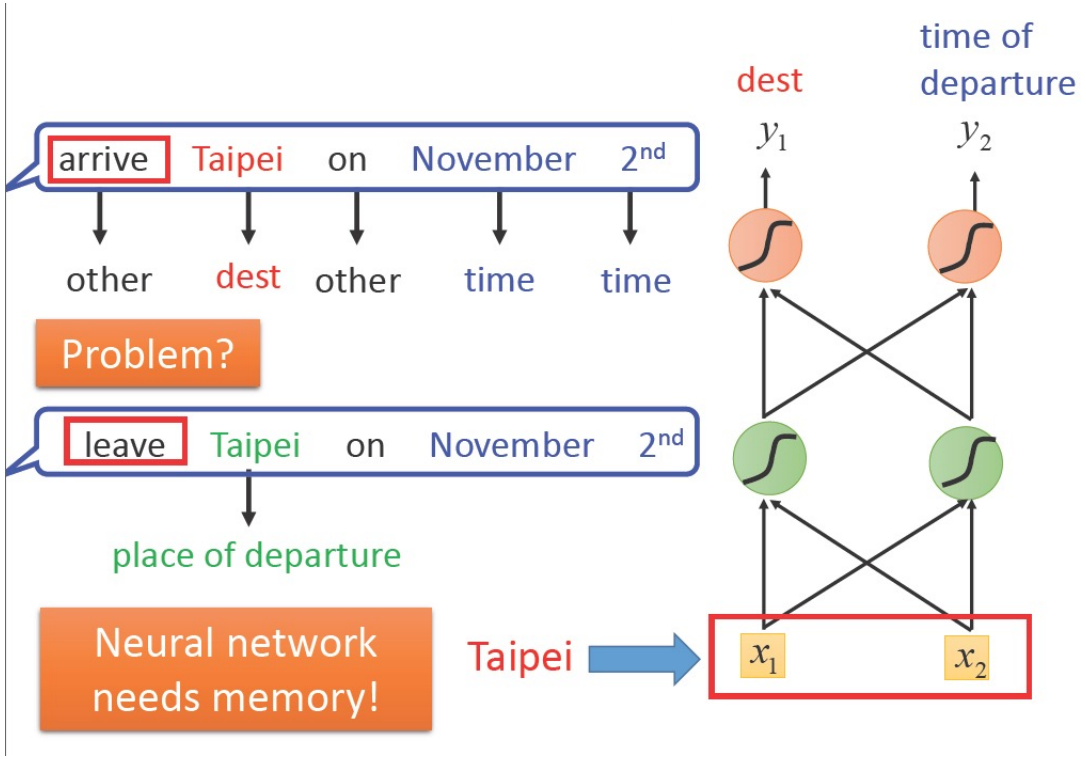
"Dog" is close to "Cat"

"Love" is close to "Hate"

"Korea" is close to "America"

Why RNN?

- Position of words is important
- Slightly change a word may change meaning of whole sentence.
- Lack of sequence concept in ordinary NN.



Memory is important

Input:
2 dimensions

x^1	x^2	x^3
4	4	1
7	7	1

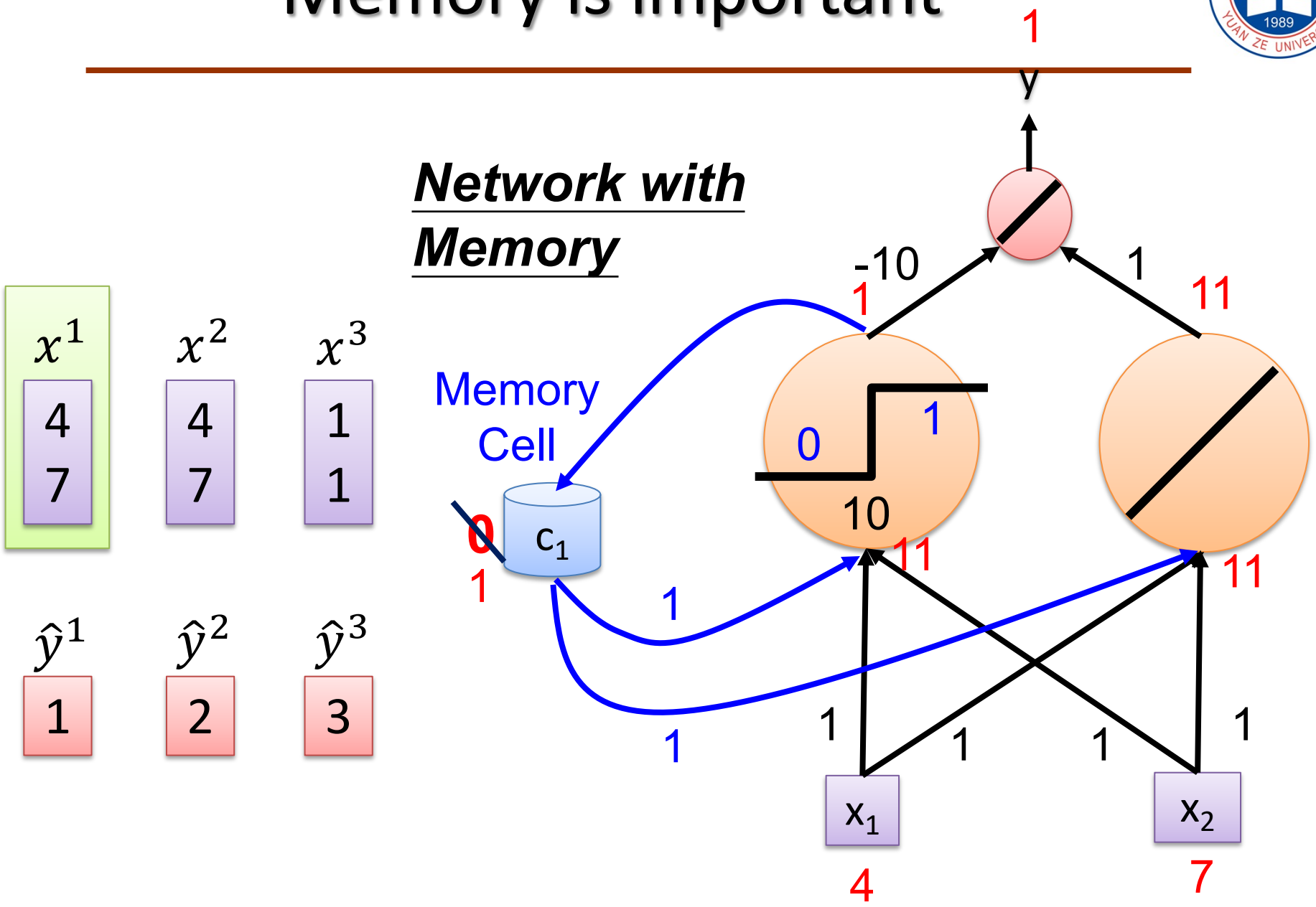
Output:
1 dimension

$\hat{y}^1$	$\hat{y}^2$	$\hat{y}^3$
1	2	3

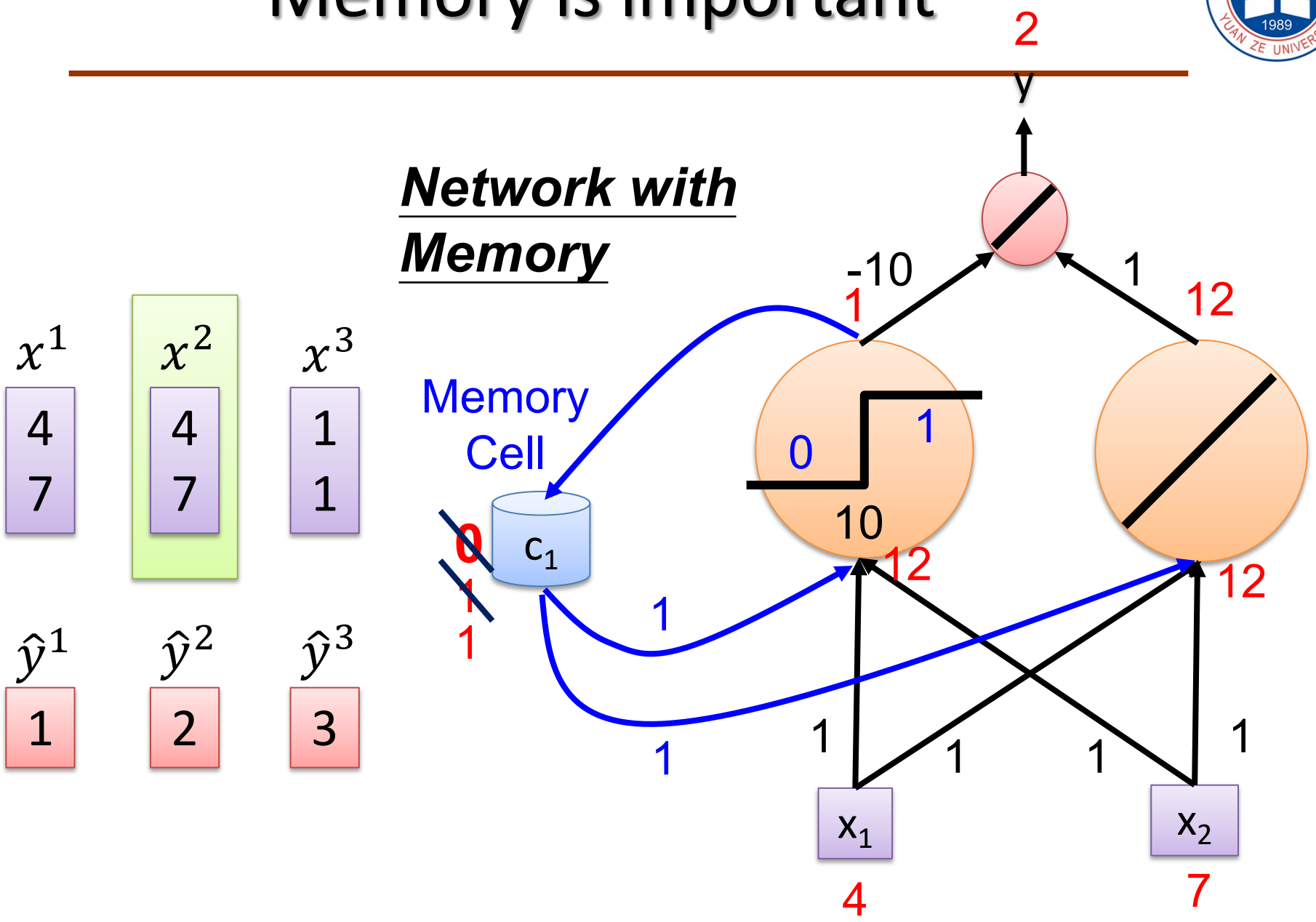
	1	1	
	1	4	4
+	1	7	7
	3	2	1

Network needs
memory to achieve this

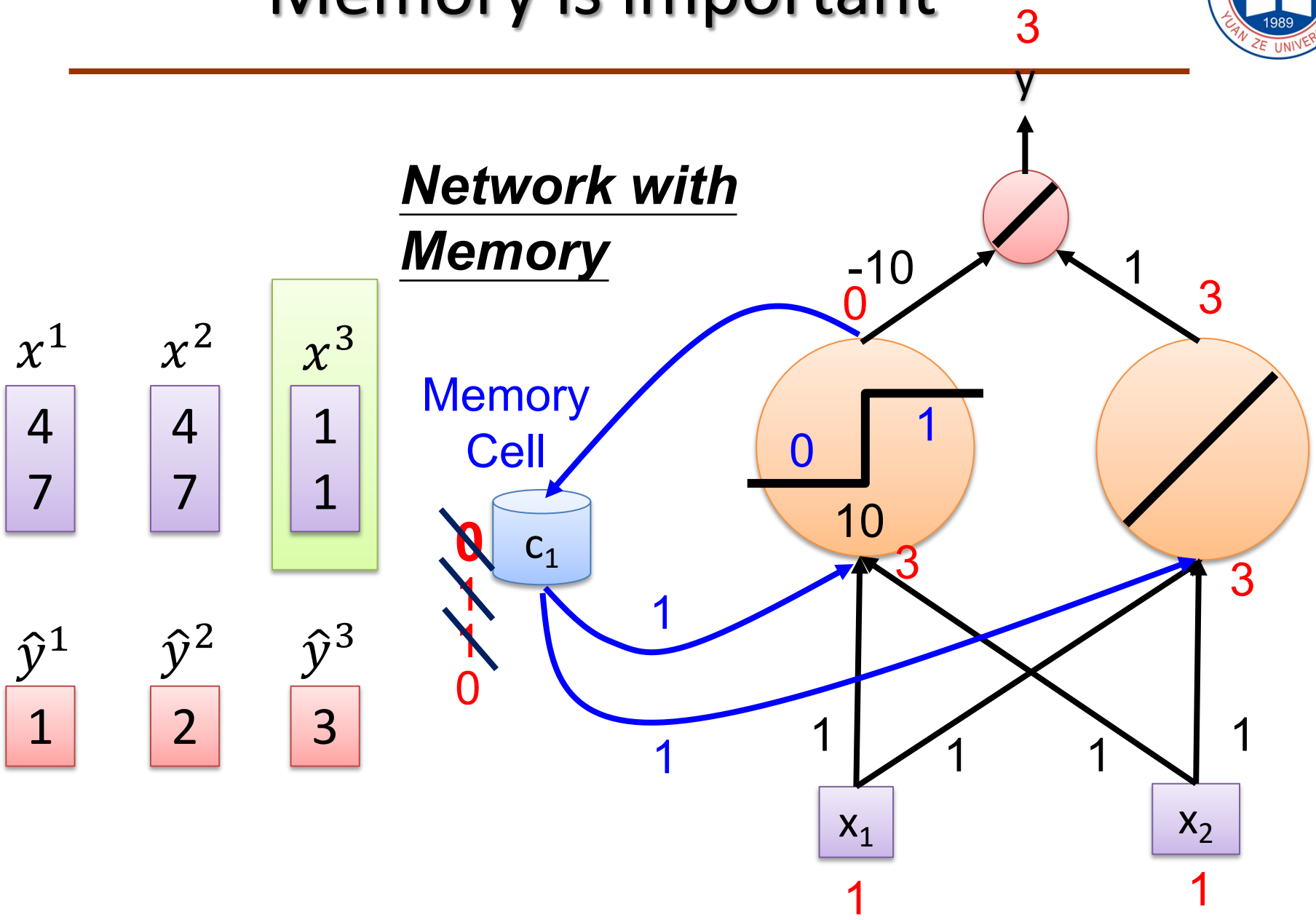
Memory is important



Memory is important

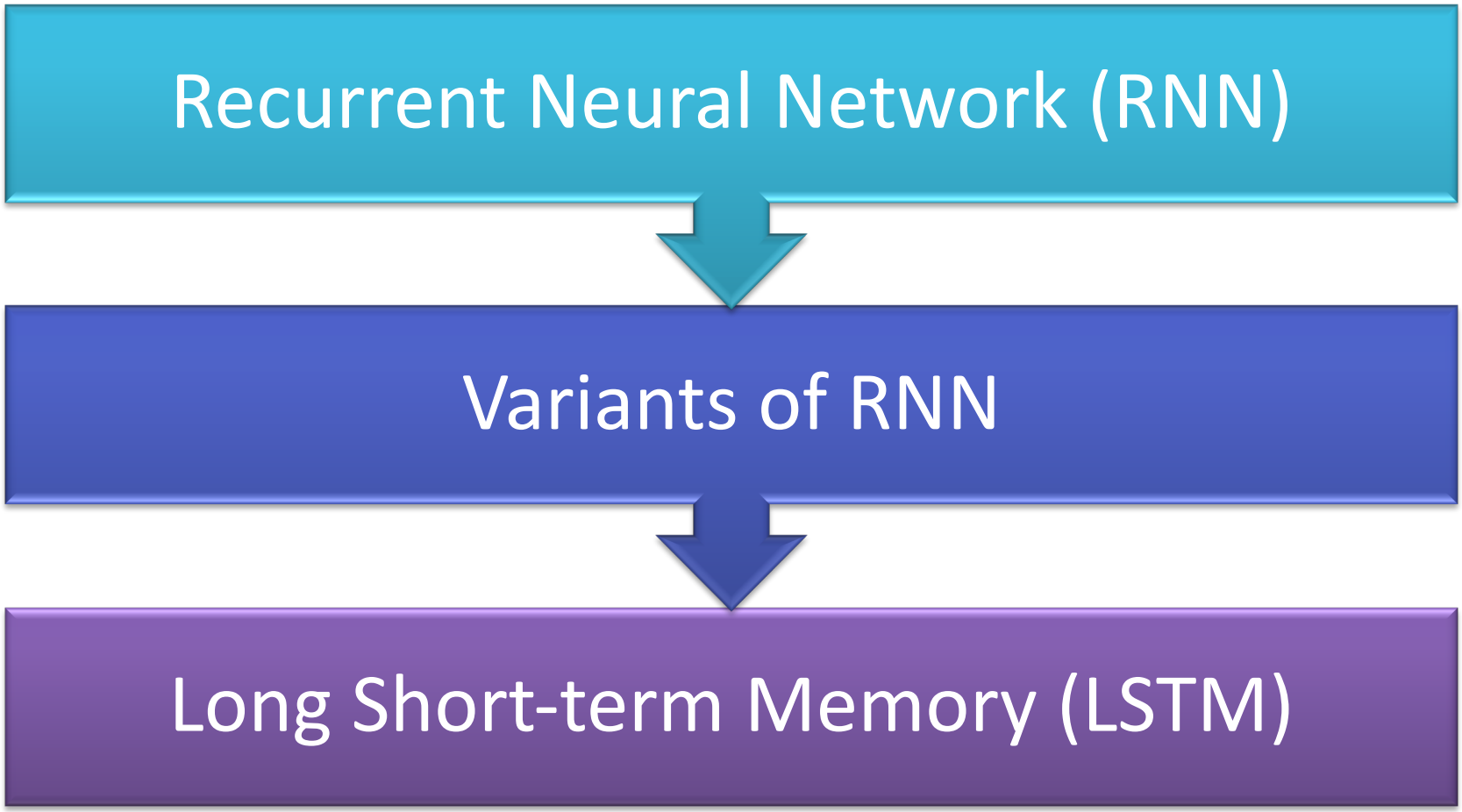


Memory is important

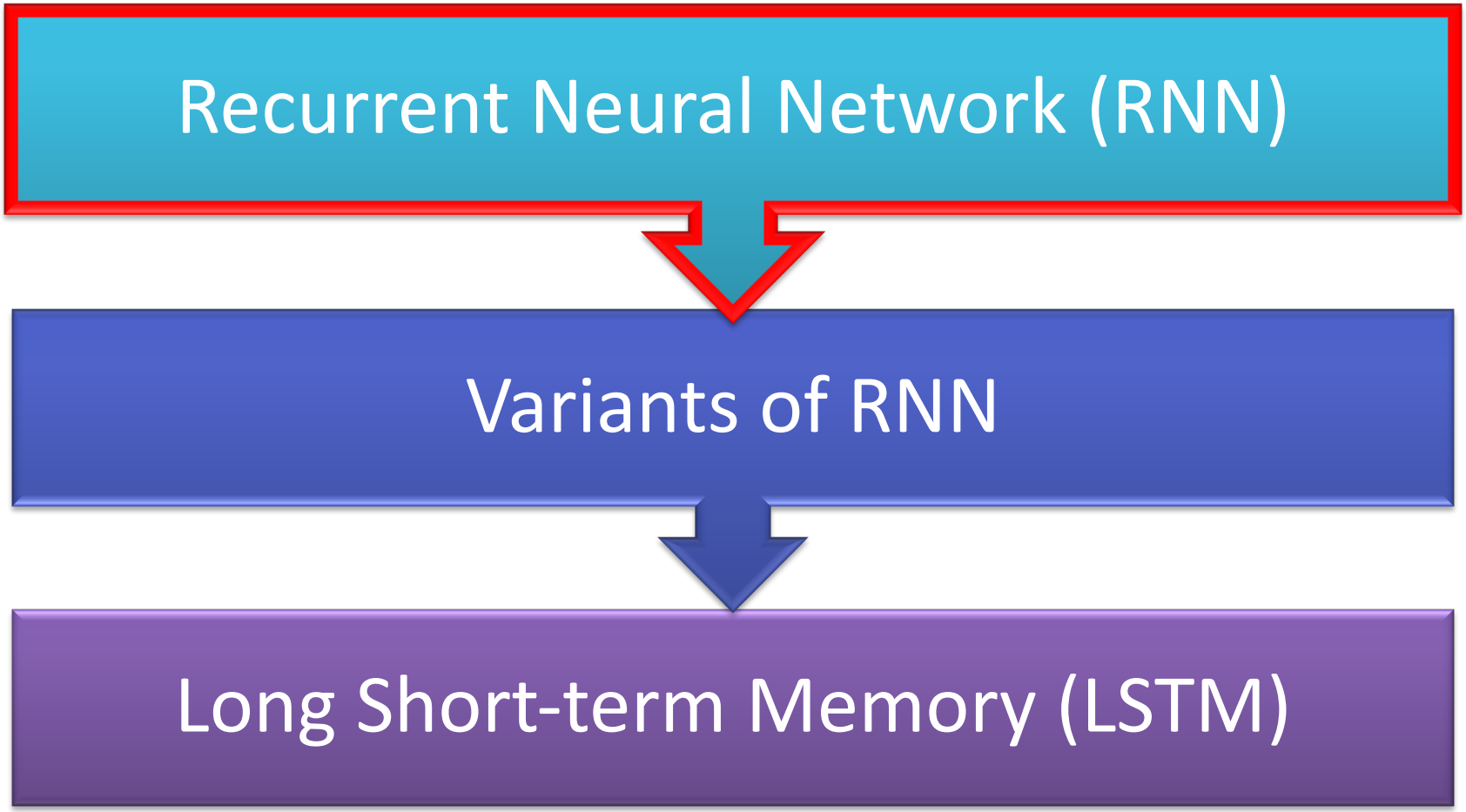


RNN Model

Outline

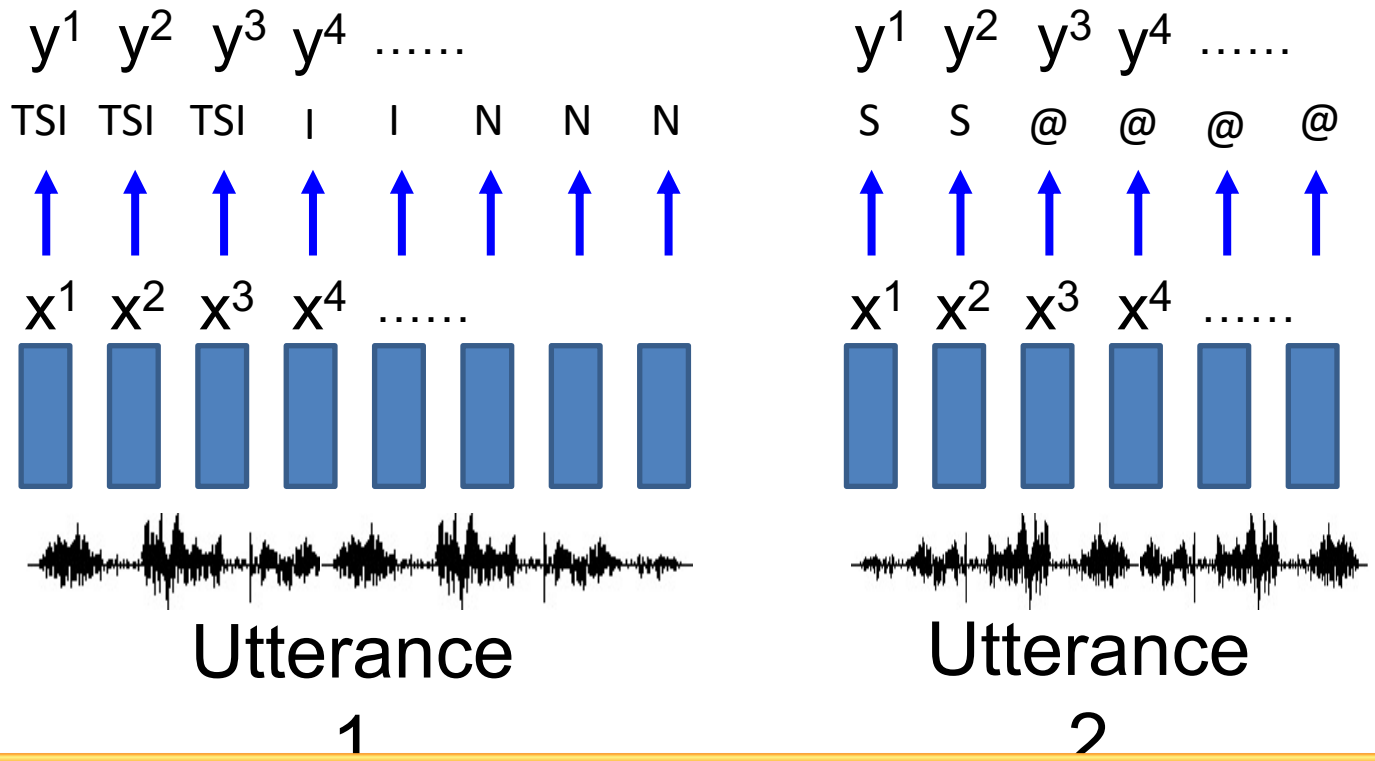

Recurrent Neural Network (RNN)
Variants of RNN
Long Short-term Memory (LSTM)

Outline


Recurrent Neural Network (RNN)
Variants of RNN
Long Short-term Memory (LSTM)

Application

- (Simplified) Speech Recognition



We use DNN. All the frames are considered independently.

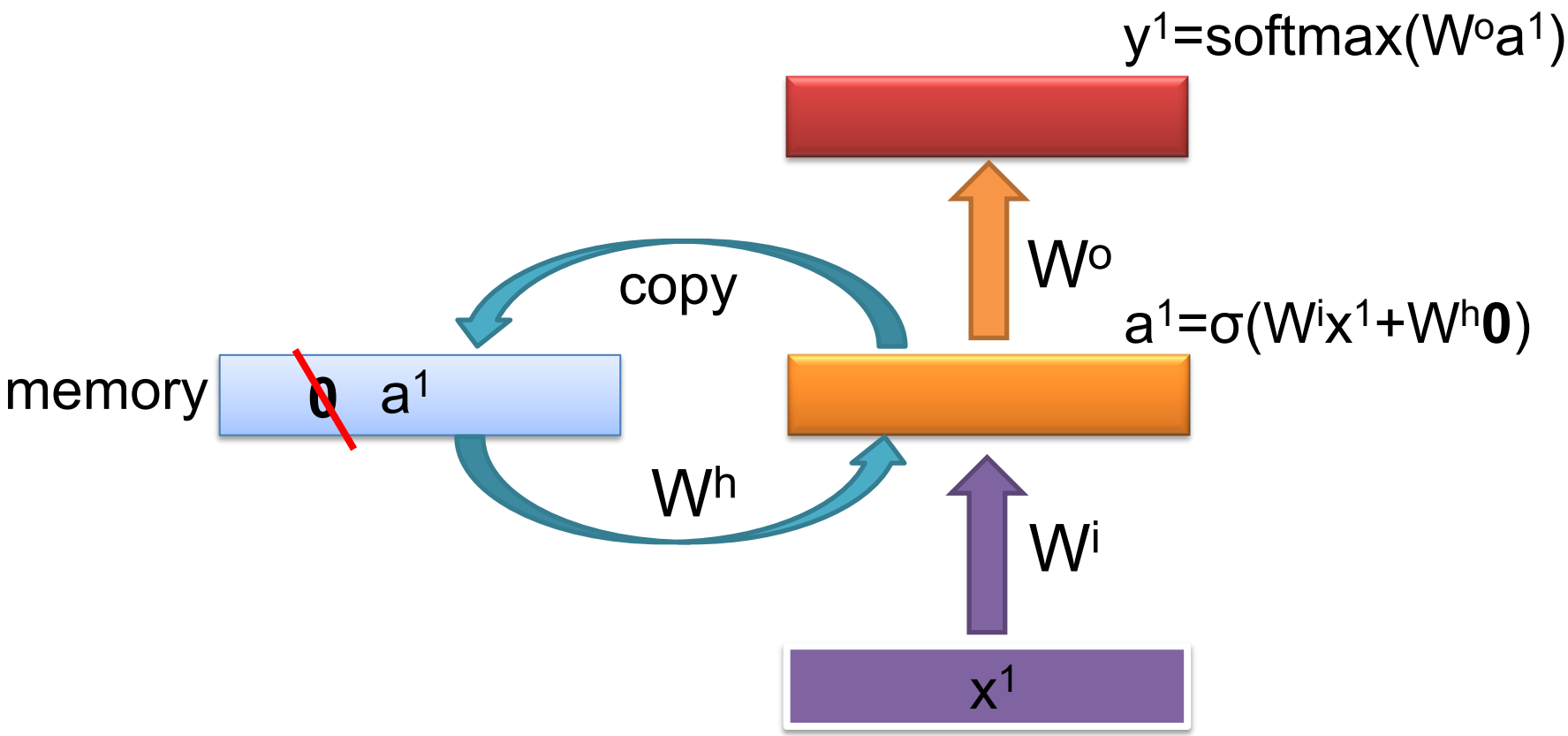
RNN

The order cannot change.

RNN input: x^1 x^2 x^3 …… x^N

Input of RNN is one utterance

y^1



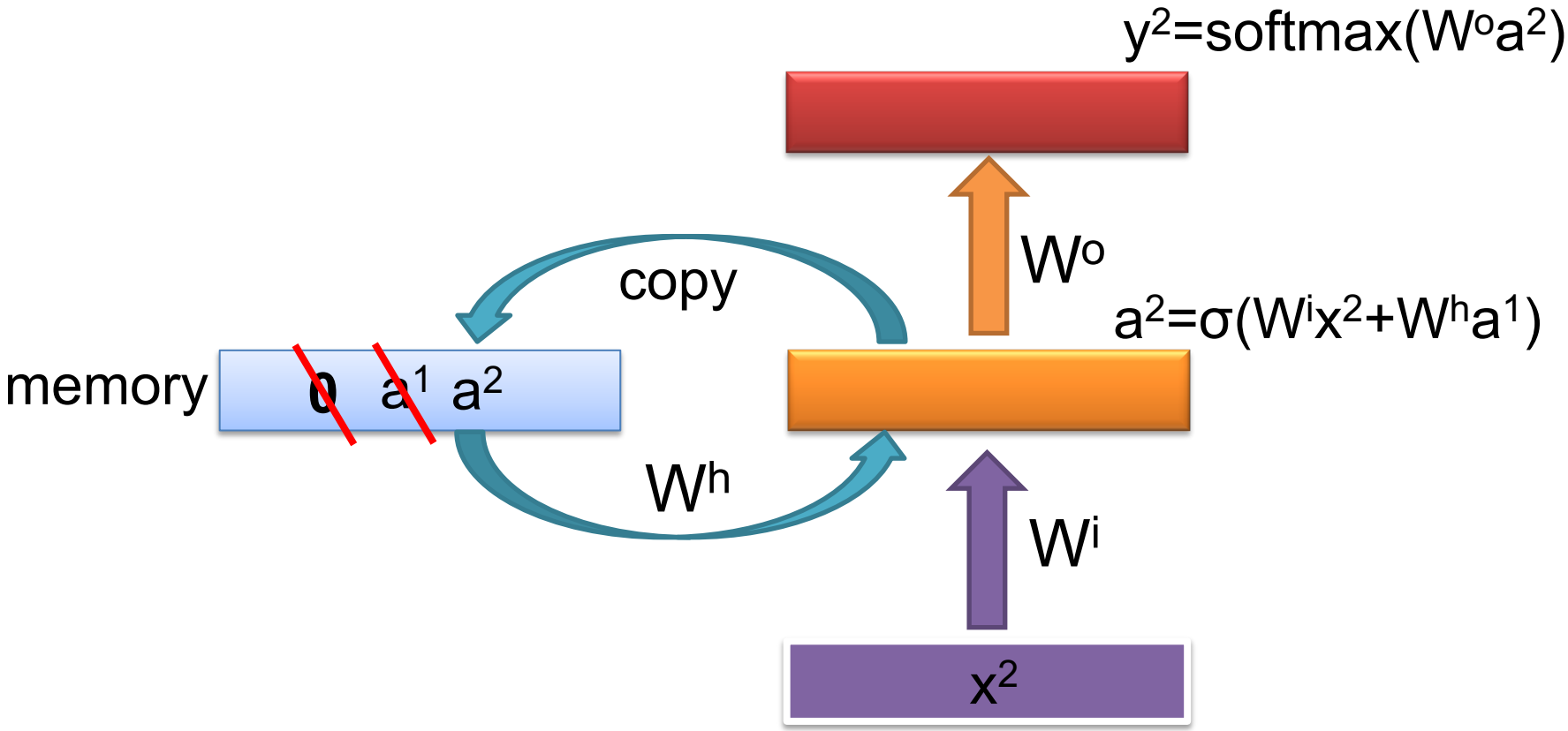
RNN

The order cannot change.

RNN input: x^1 x^2 x^3 …… x^N

y^1 y^2

Input of RNN is one utterance



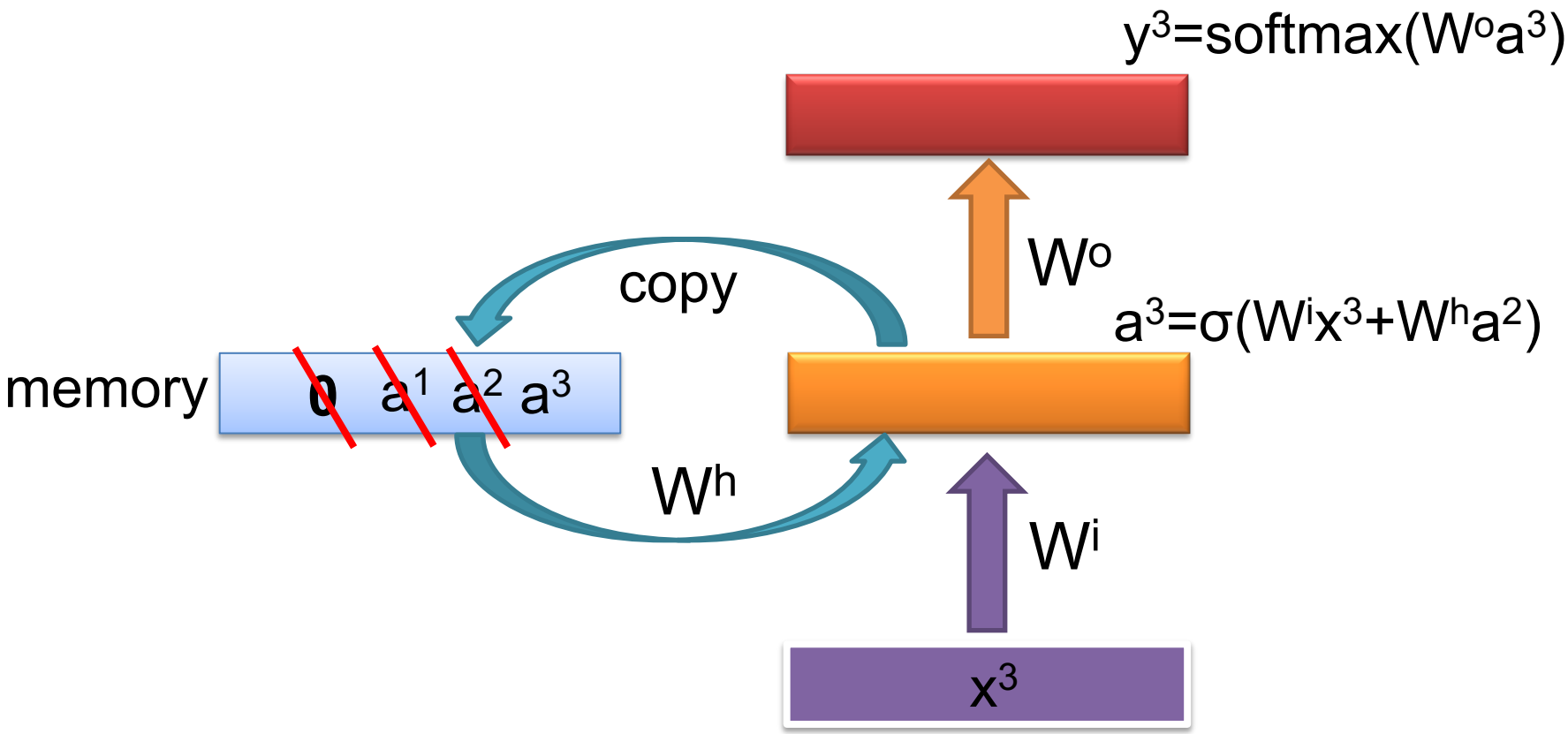
RNN

The order cannot change.

RNN input: x^1 x^2 x^3 …… x^N

y^1 y^2 y^3

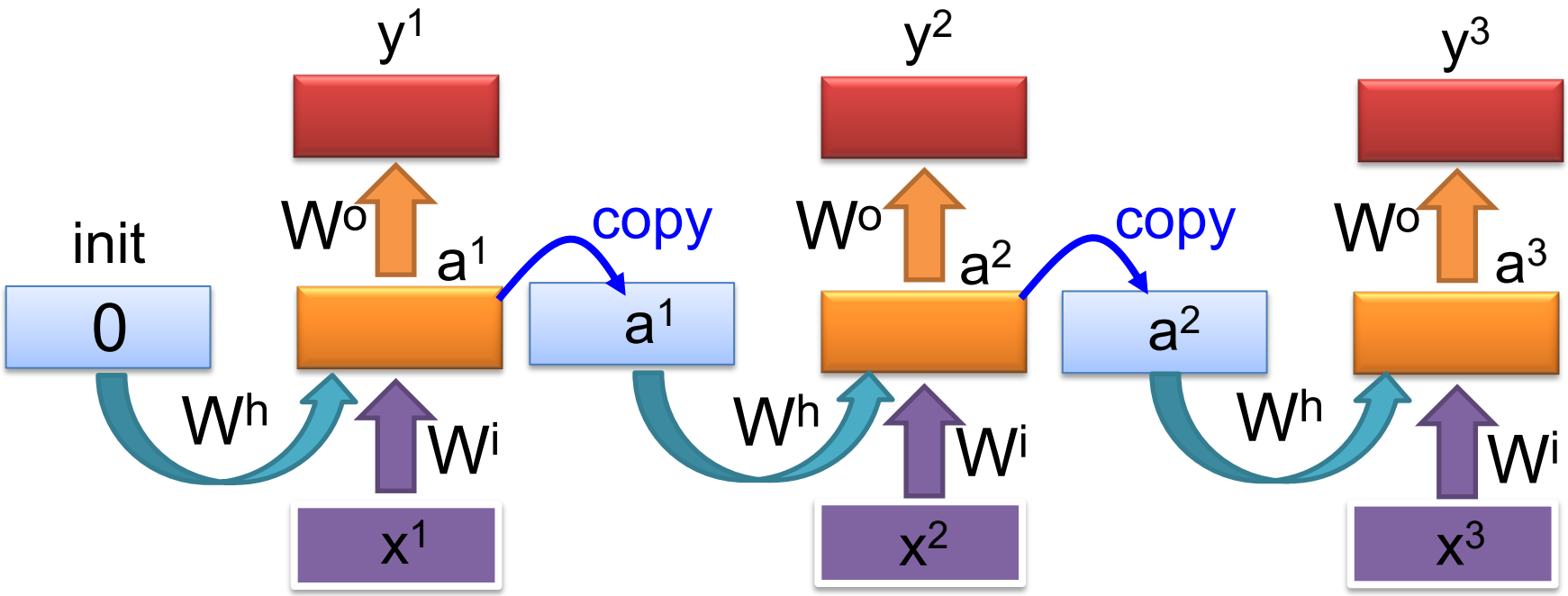
Input of RNN is one utterance



RNN

Input data: x^1 x^2 x^3 …… x^N

Input of RNN is one utterance



The same network is used again and again.

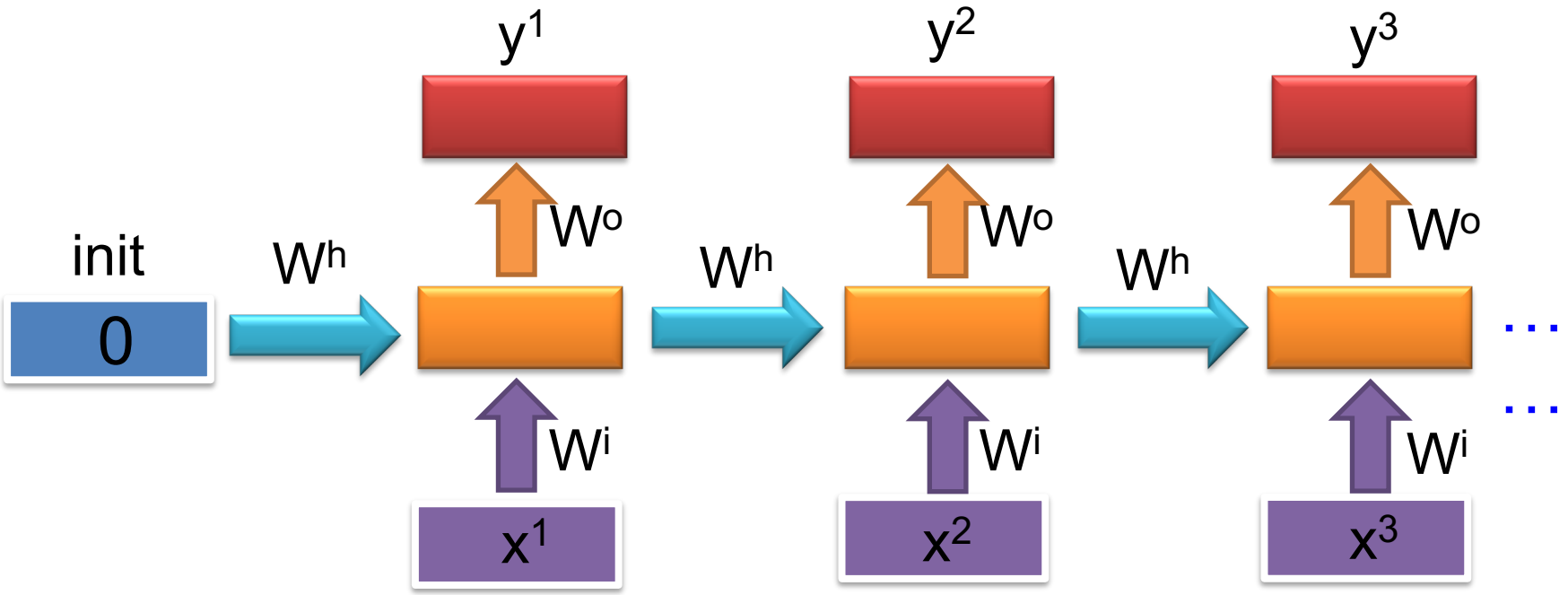
The values store in memory is different.

Output y^i depends on x^1, x^2, …… x^i

RNN

Input data: x^1 x^2 x^3 …… x^N

Input of RNN is one utterance



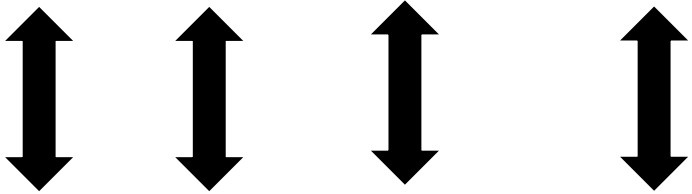
The same network is used again and again.

Output y^i depends on x^1, x^2, …… x^i

Cost

RNN input: x^1 x^2 x^3 …… x^N

RNN output: y^1 y^2 y^3 …… y^N

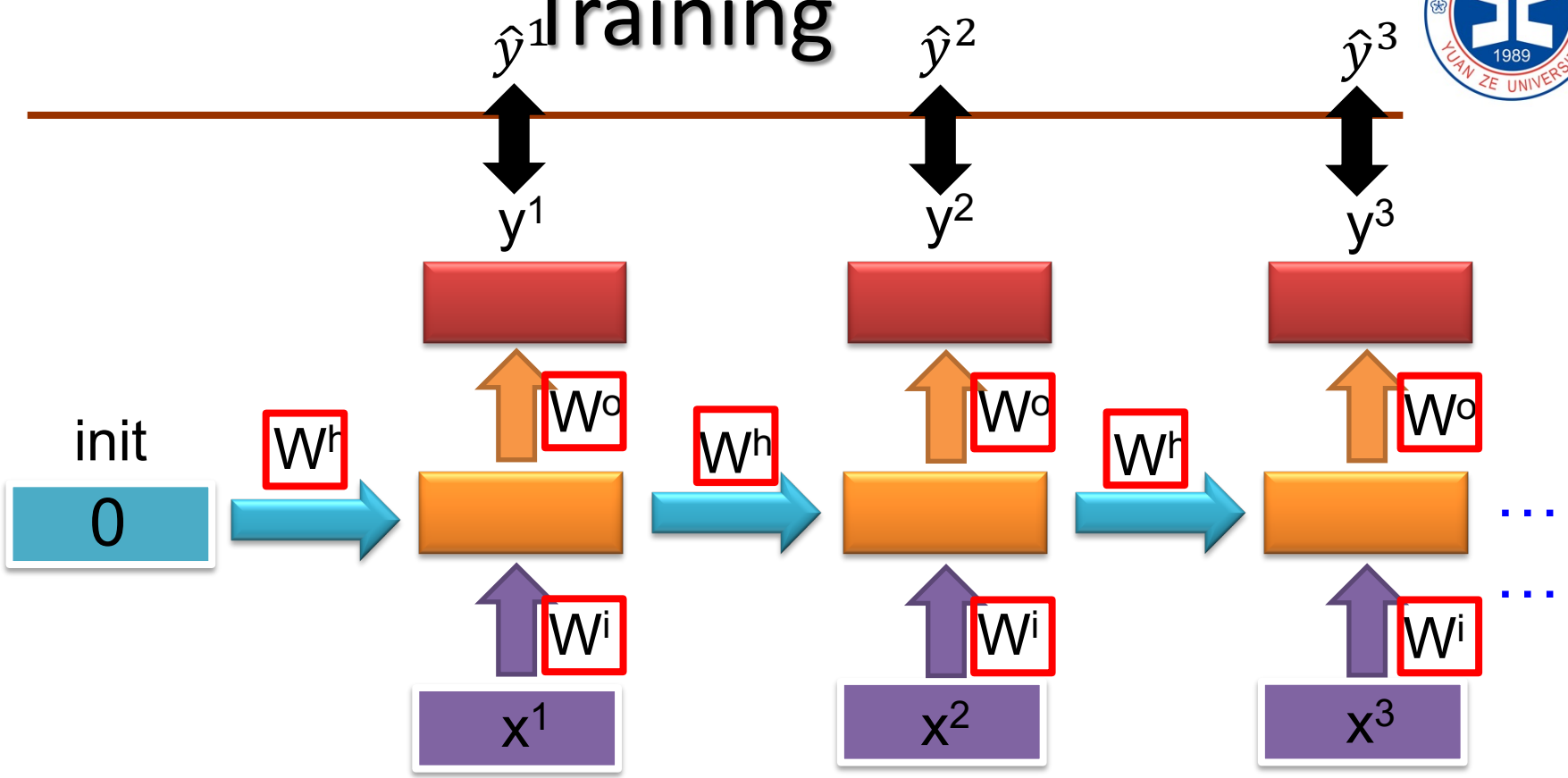


RNN output: $\hat{y}^1$ $\hat{y}^2$ $\hat{y}^3$ …… $\hat{y}^N$

$$C = \frac{1}{2}\sum_{n=1}^{N} \|y^n - \hat{y}^n\|^2$$

$$C = \frac{1}{2}\sum_{n=1}^{N} -log y_{r^n}^n$$

Training



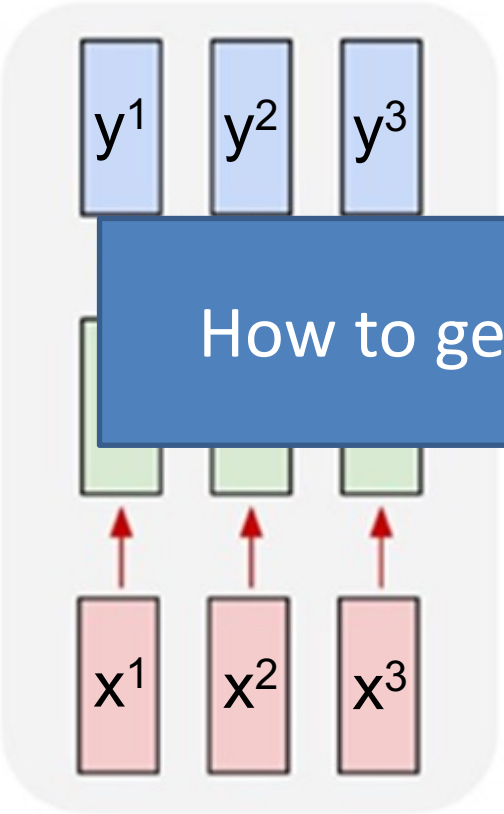
w is an element in W^h, W^i or W^o ➡ $w \leftarrow w - \eta \partial C / \partial w$

➡ Backpropagation through time (BPTT)

RNN Training is very difficult in practice.

Applications Suitable for RNN

- Input and output are vector sequences with ***the same length***



How to generate word as input?

y1 y2 y3 y4

PN V D N

John saw the saw.

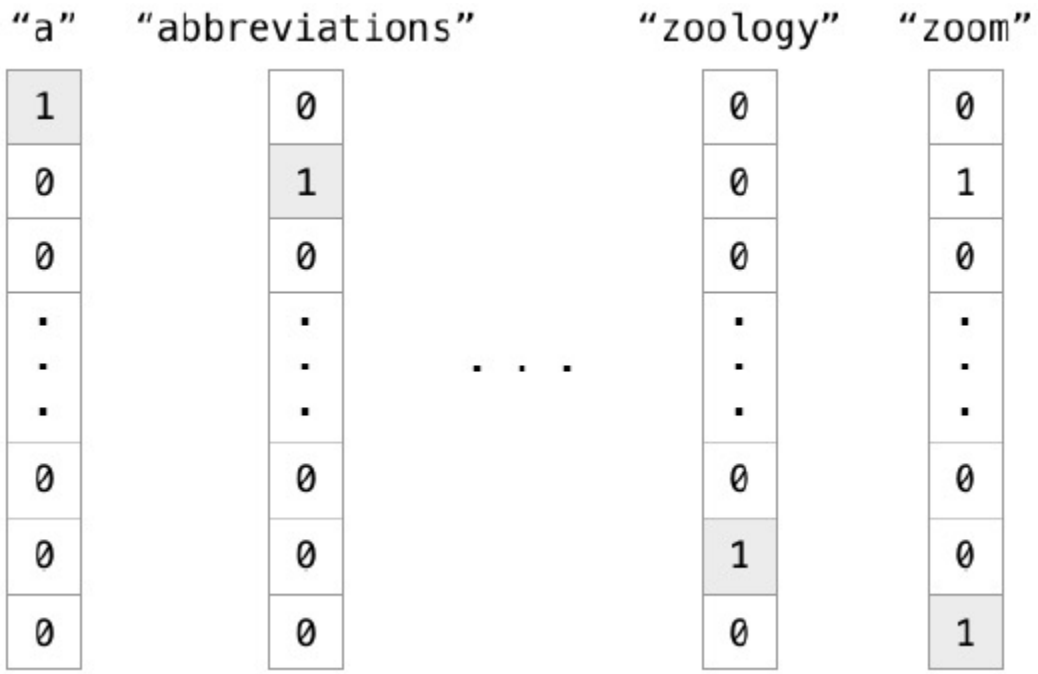
POS Tagging

More Applications

- Name entity recognition
  - Identifying names of people, places, organizations, etc. from a sentence
  - Harry Potter is a student of Hogwarts and lived on Privet Drive.
    - people, organizations, places, not a name entity
- Information extraction
  - Extract pieces of information relevant to a specific application, e.g. flight booking
  - I would like to leave Boston on November 2nd and arrive in Taipei before 2 p.m.
    - place of departure, destination, time of departure, time of arrival, other

Word Representation: 1-of-N Encoding

- We let machine read a lot of articles and use one hot encoding on each word.



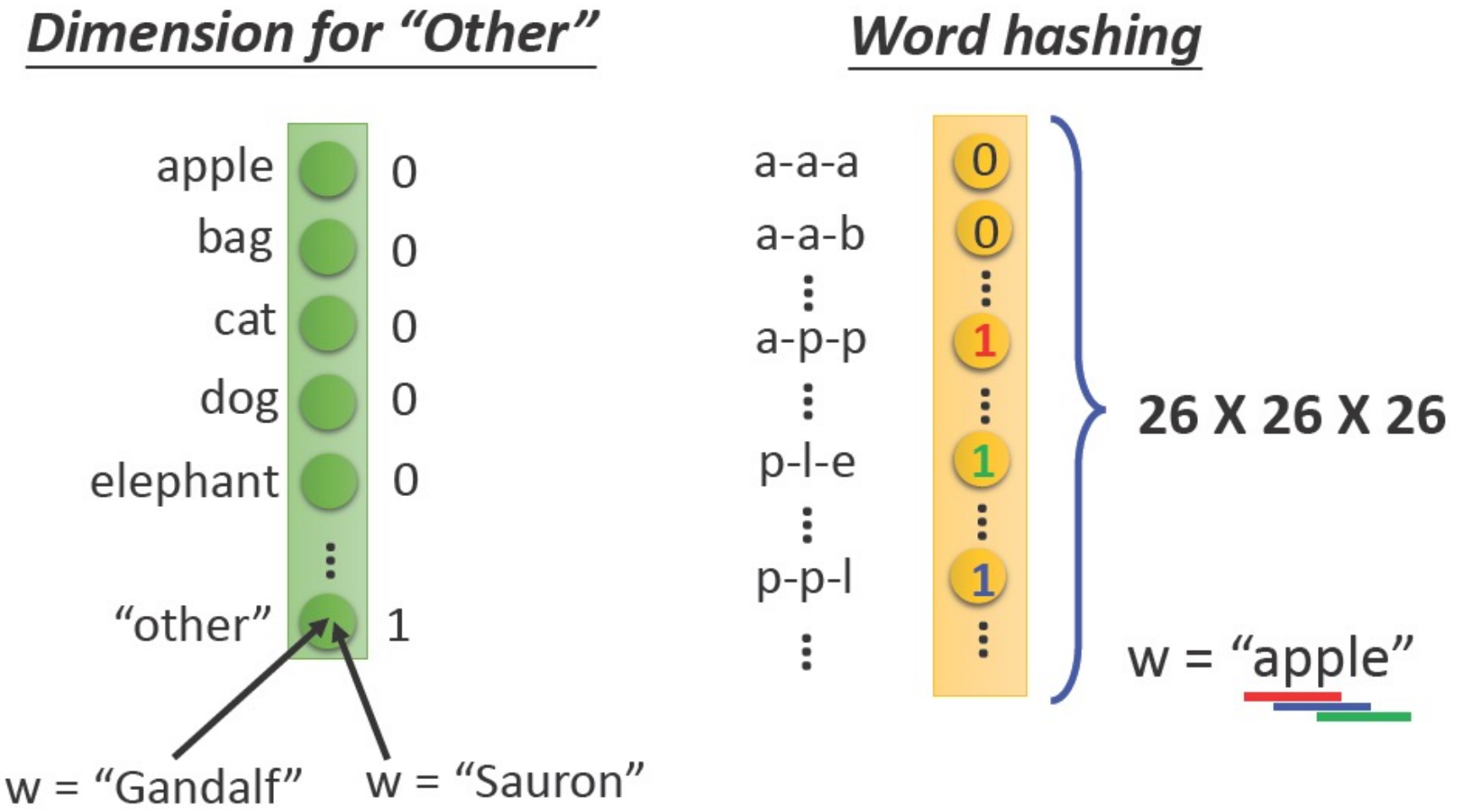
Drawback of 1-of-N Encoding

- Dog is not close to cat
  - Machine can not understand the meaning of word.
- Waste a lot of entity
  - Most of entity are zero
  - Dimension of vector = vocabulary size (very large)

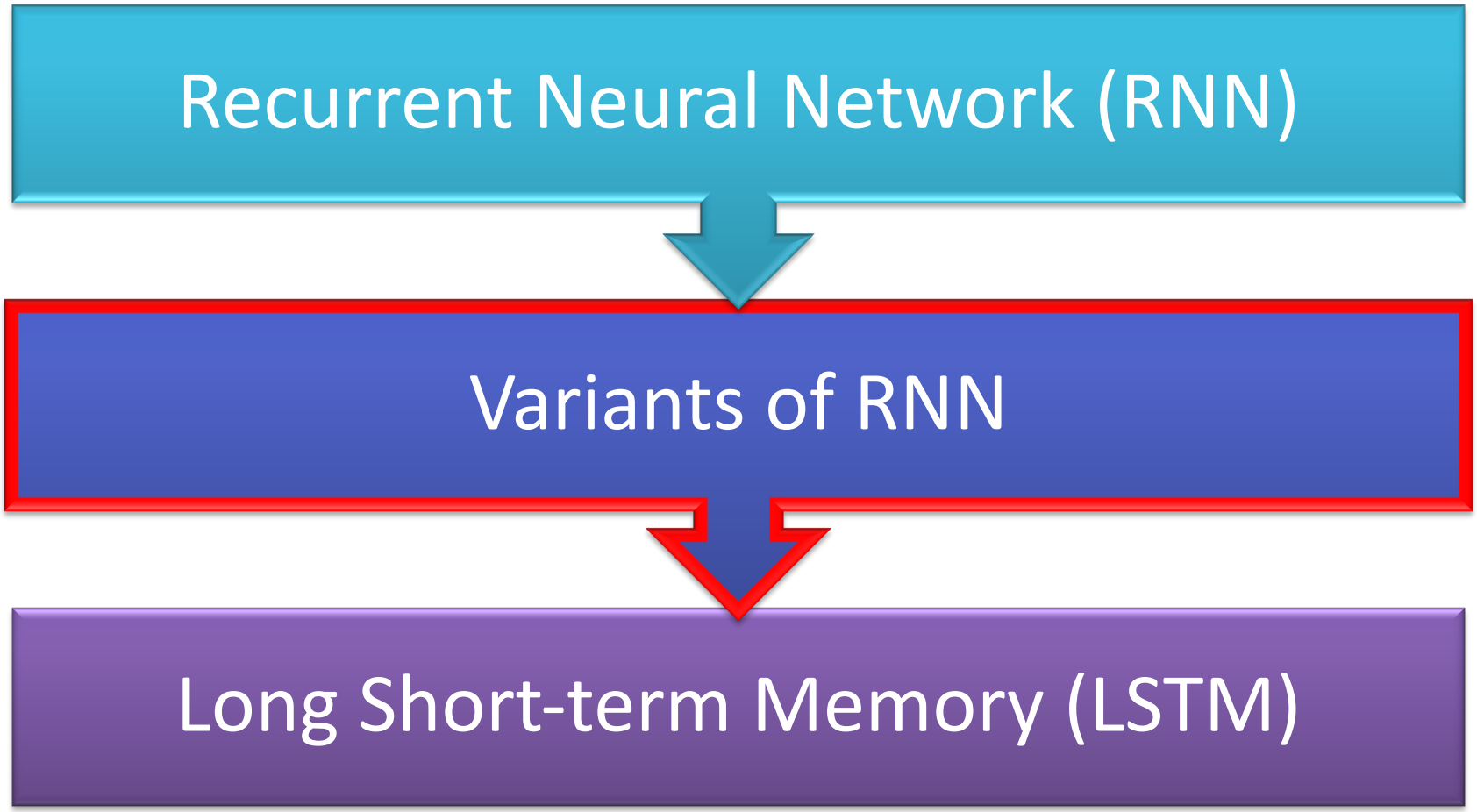
Drawback of 1-of-N Encoding

- Dog is not close to cat
  - Machine can not understand the meaning of word.
- Waste a lot of entity
  - Most of entity are zero
  - Dimension of vector = vocabulary size (very large)

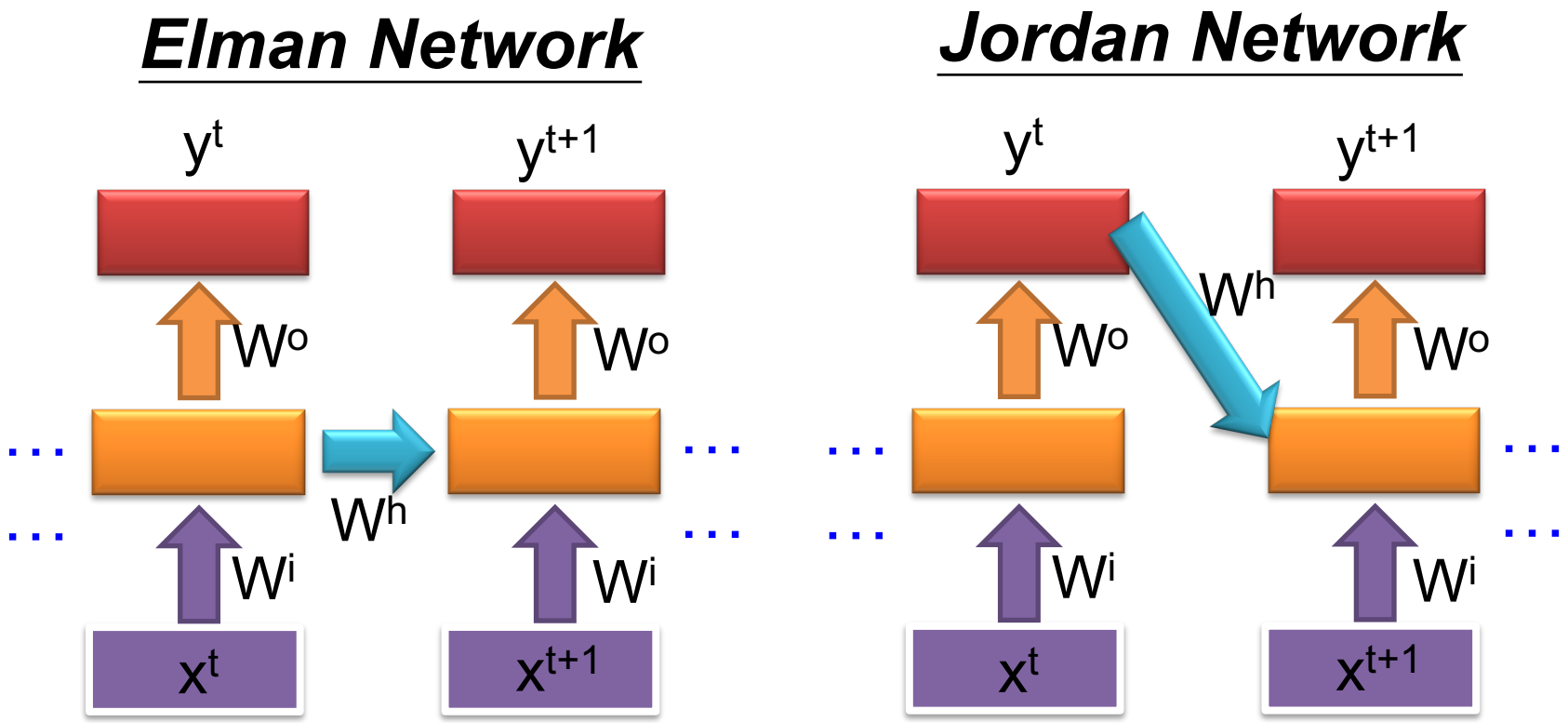
Beyond 1-of-N Encoding



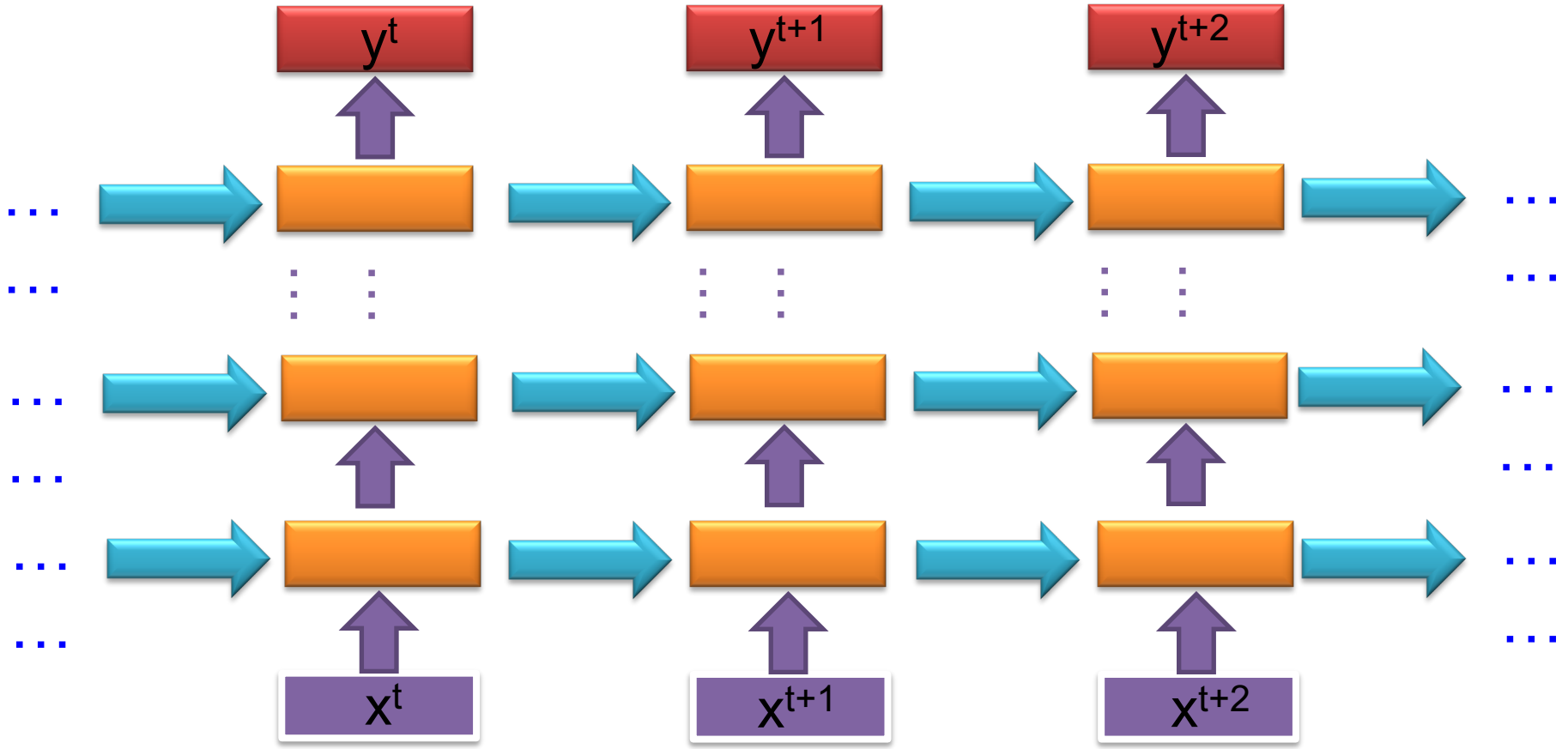
Outline


Recurrent Neural Network (RNN)
Variants of RNN
Long Short-term Memory (LSTM)

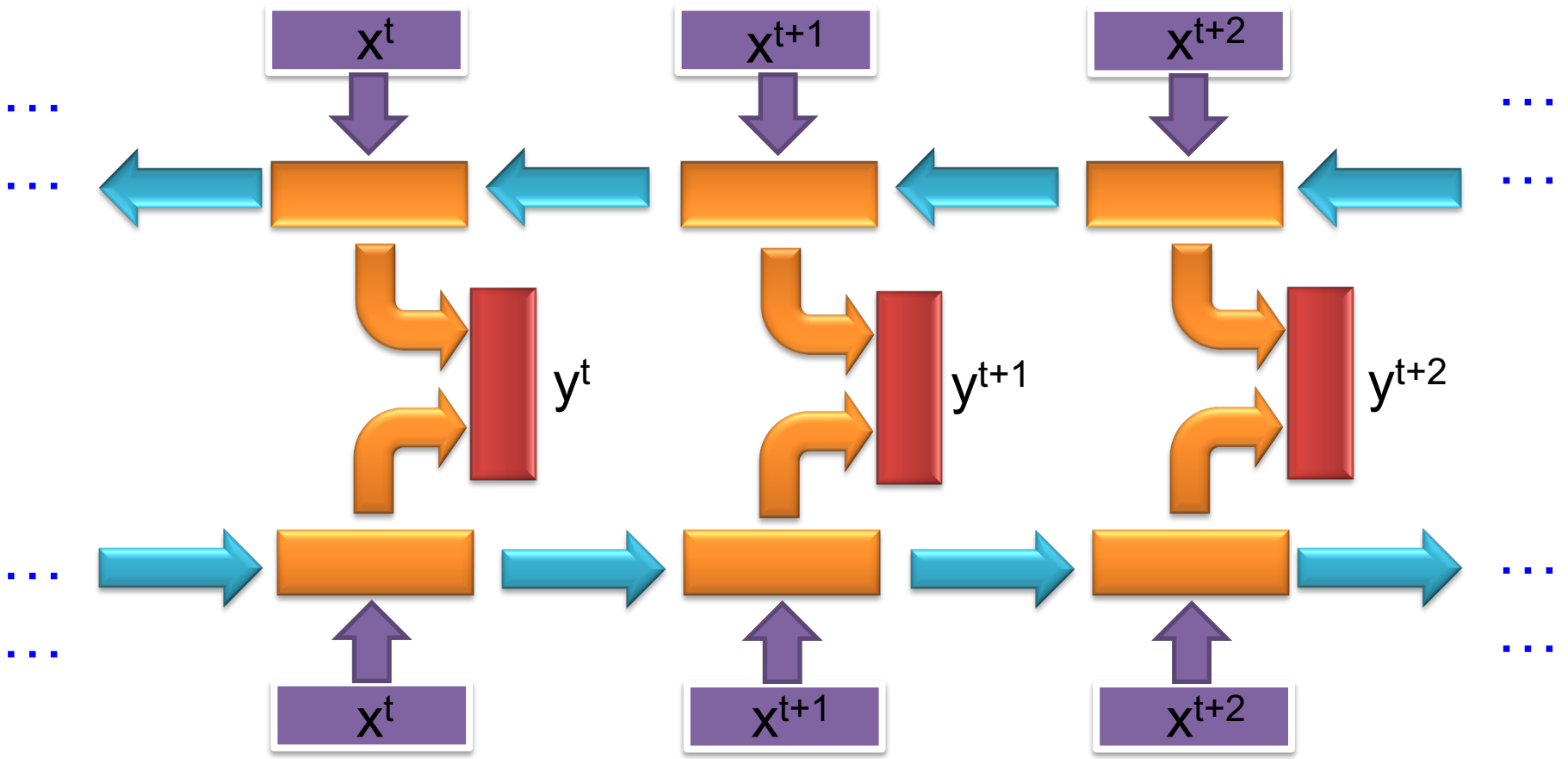
Elman Network & Jordan Network



Deep RNN



Bidirectional RNN



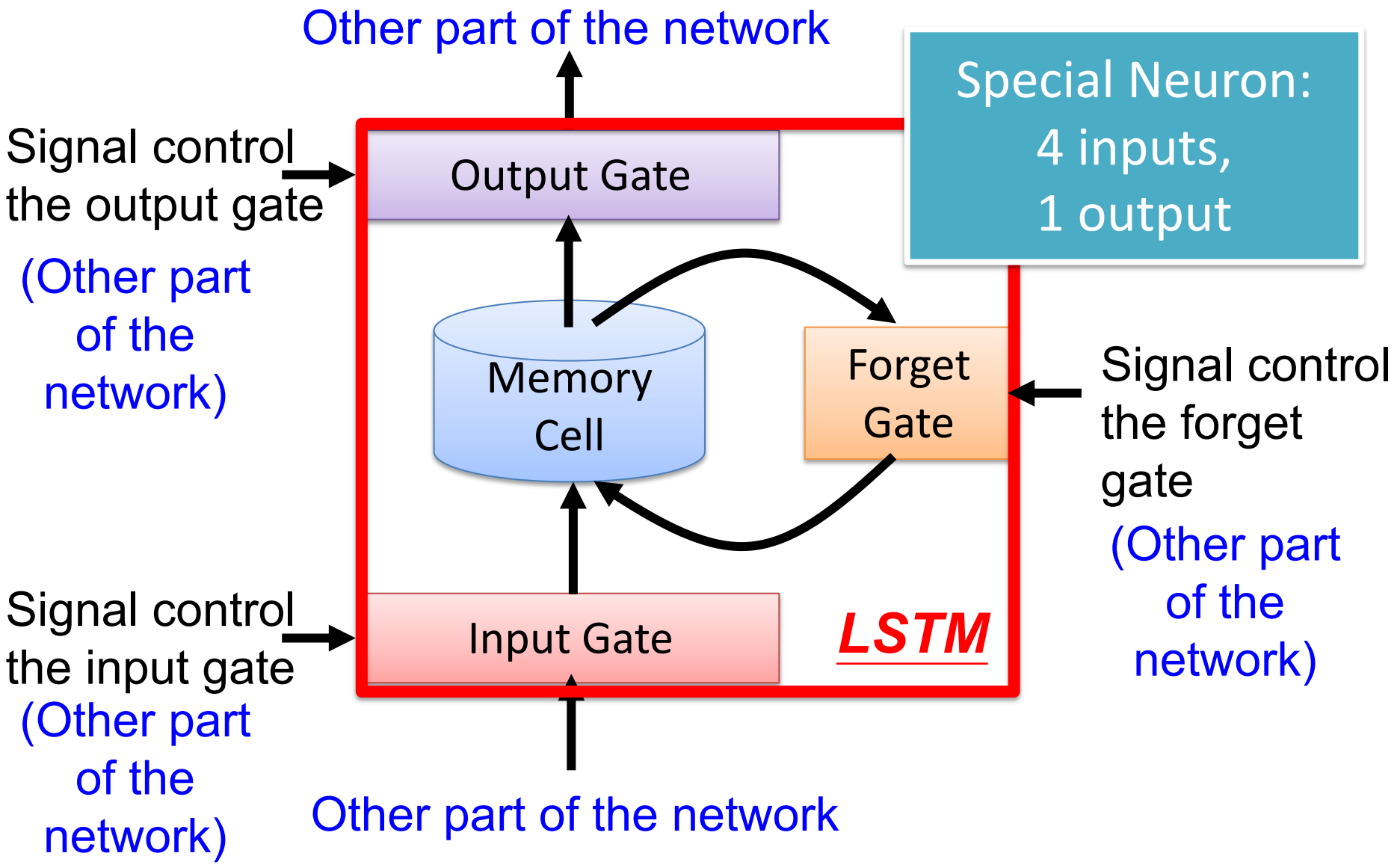
Outline

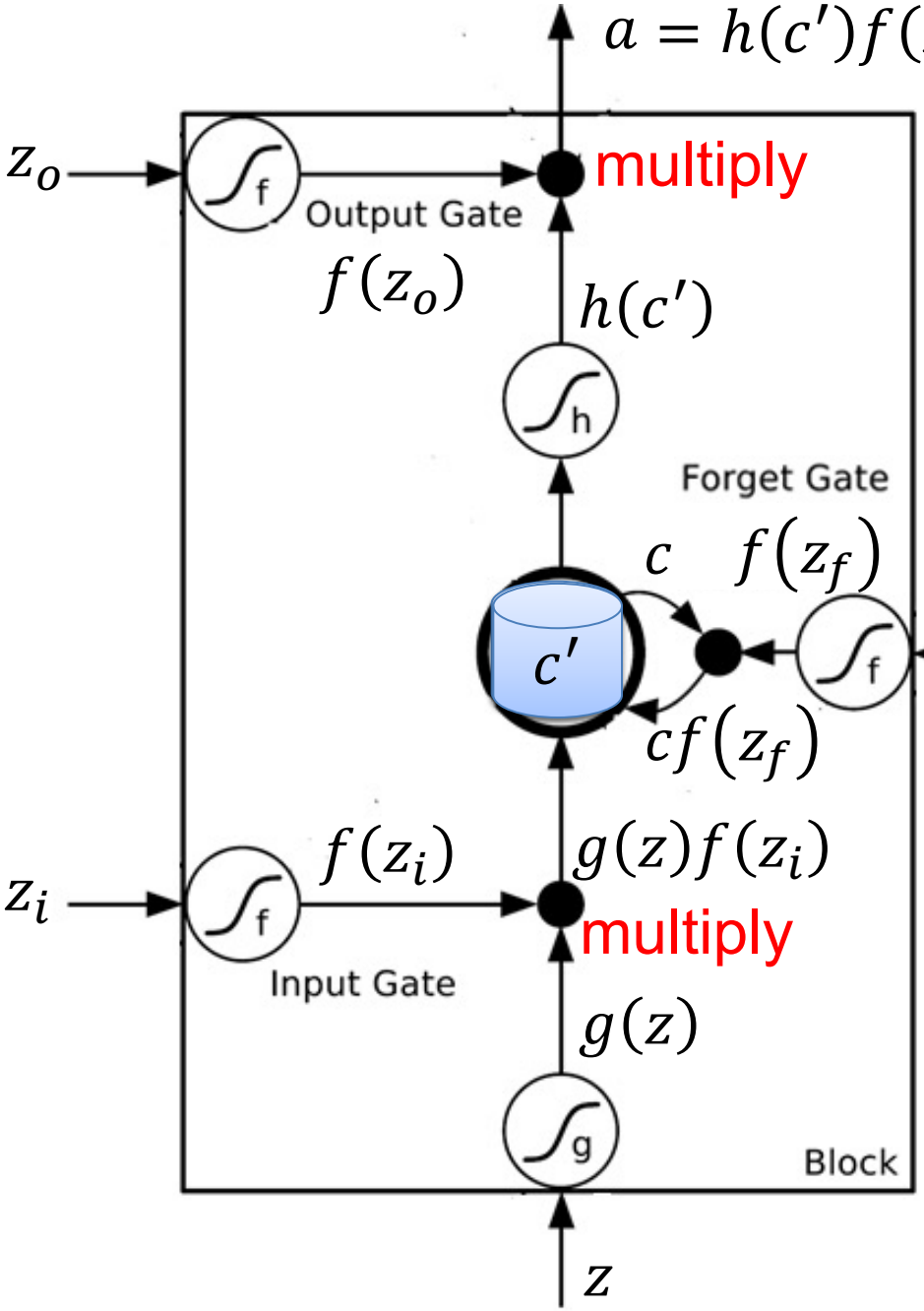
Recurrent Neural Network (RNN)

Variants of RNN

Long Short-term Memory (LSTM)

Long Short-term Memory (LSTM)





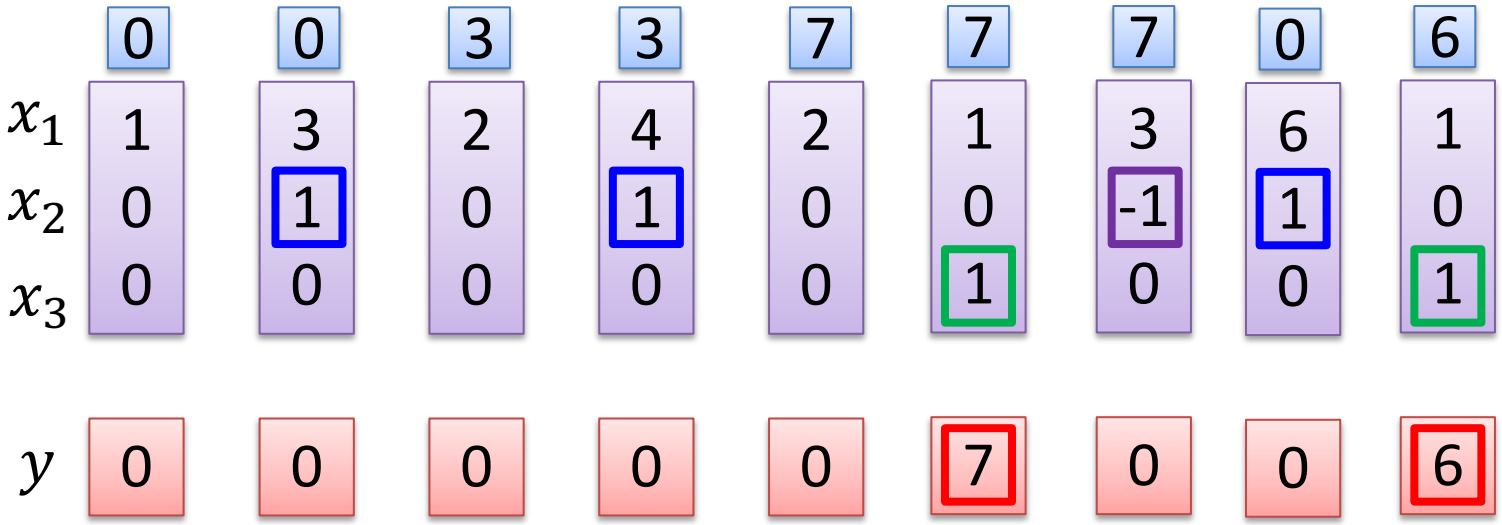
Activation function f is usually a sigmoid function

Sigmoid function between 0 and 1

Mimic open and close gate

$$c' = g(z)f(z_i) + cf(z_f)$$

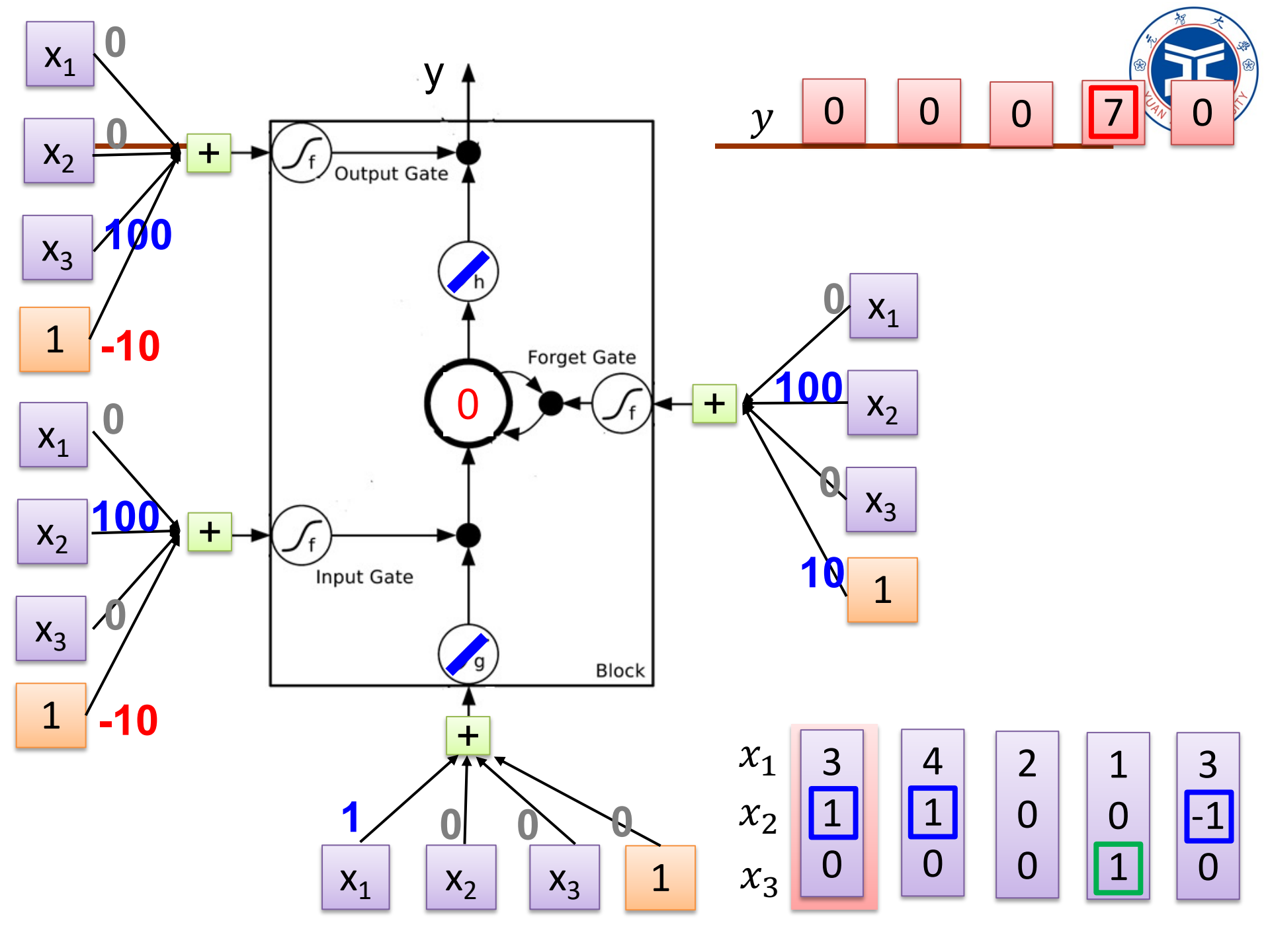
LSTM - Example

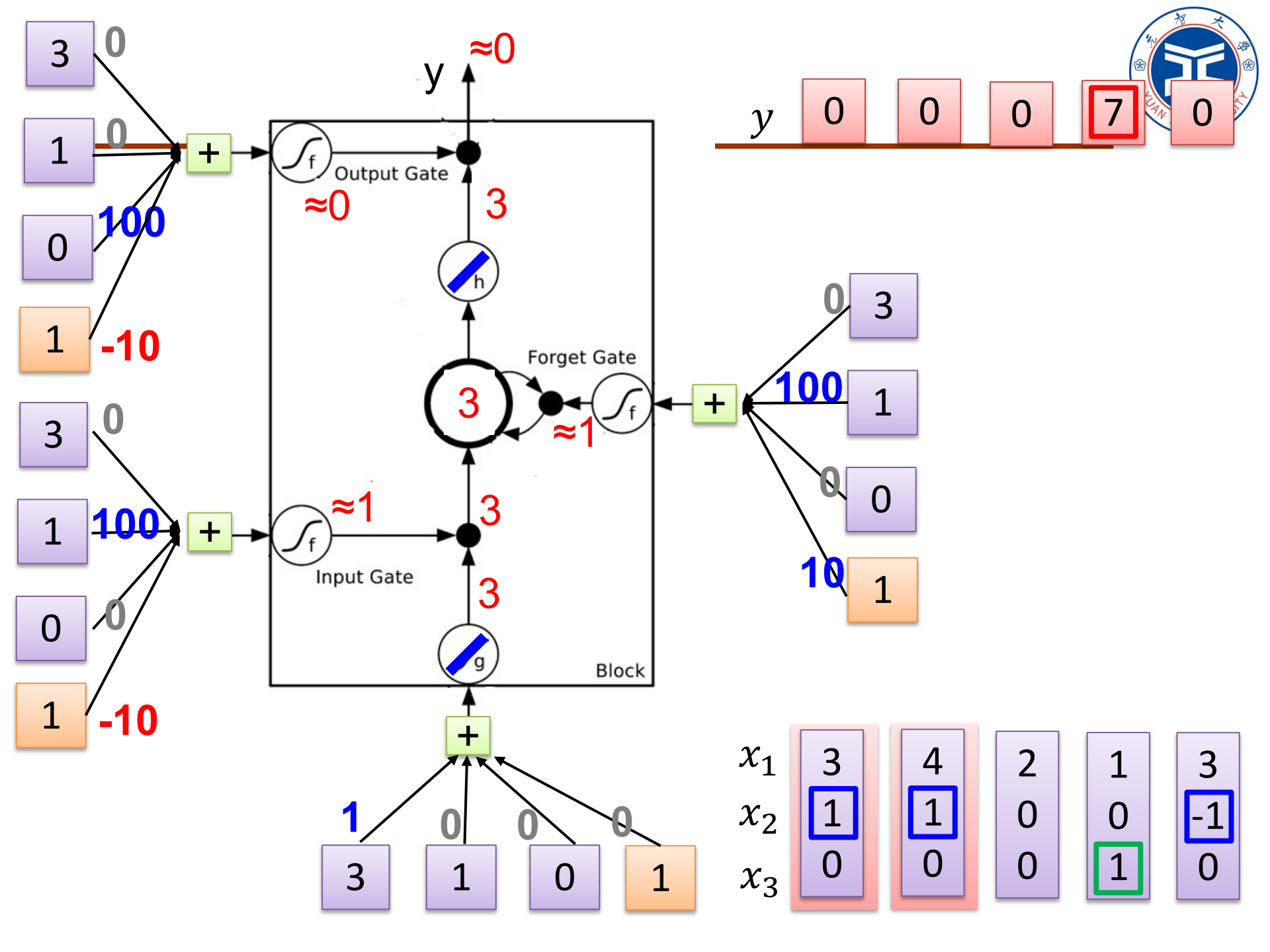


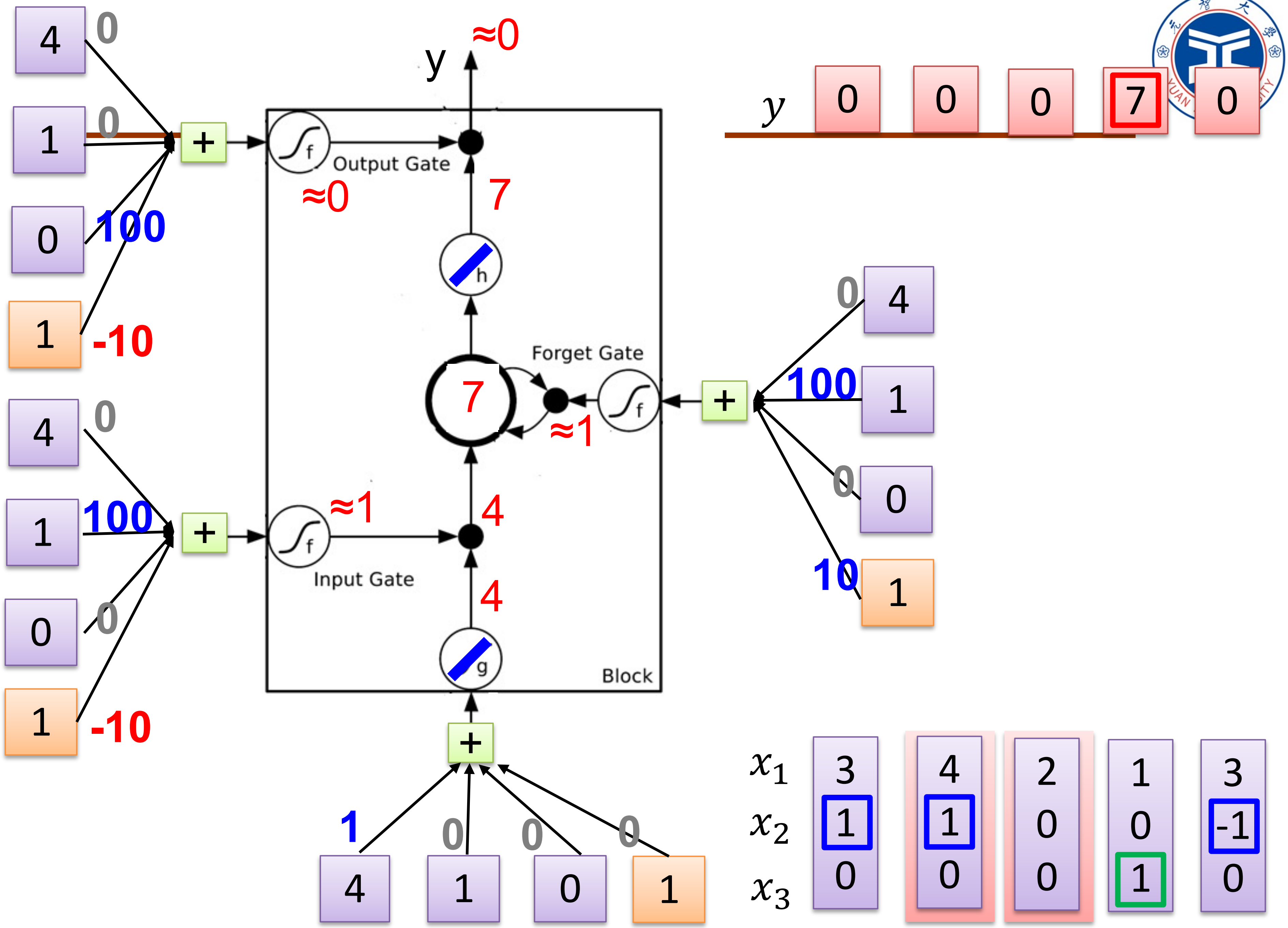
When x_2 = 1, add the numbers of x_1 into the memory

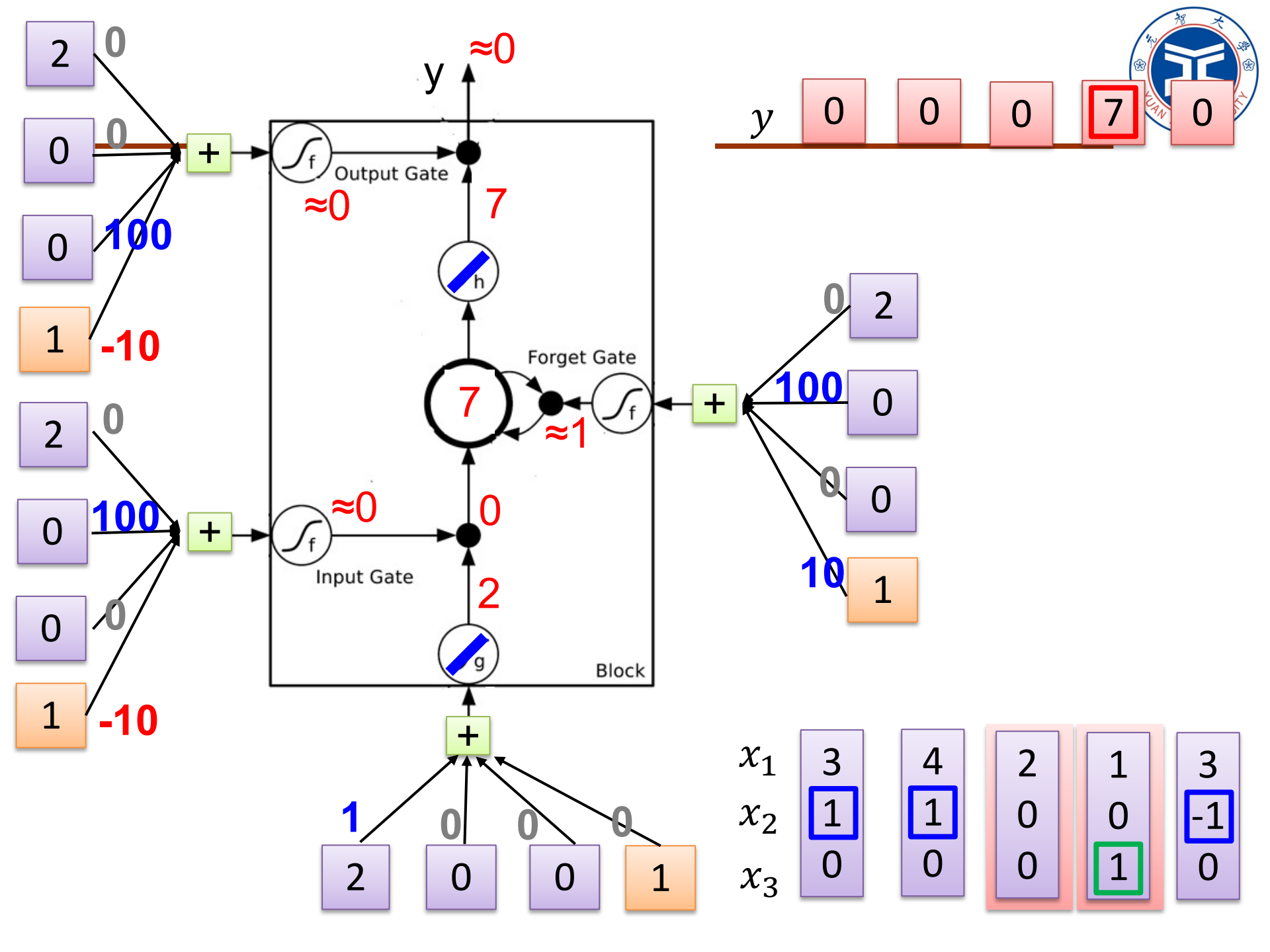
When x_2 = -1, reset the memory

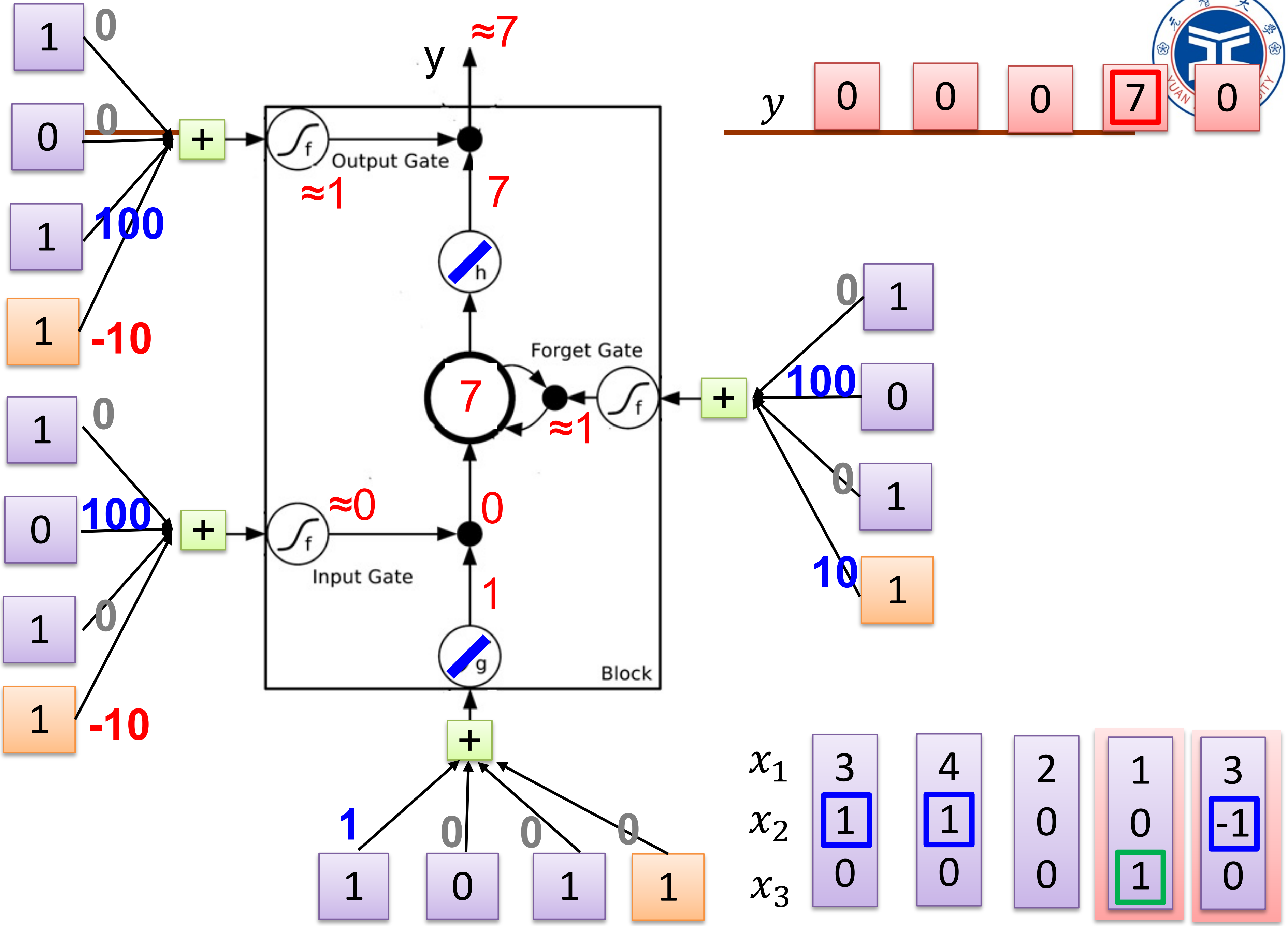
When x_3 = 1, output the number in the memory.

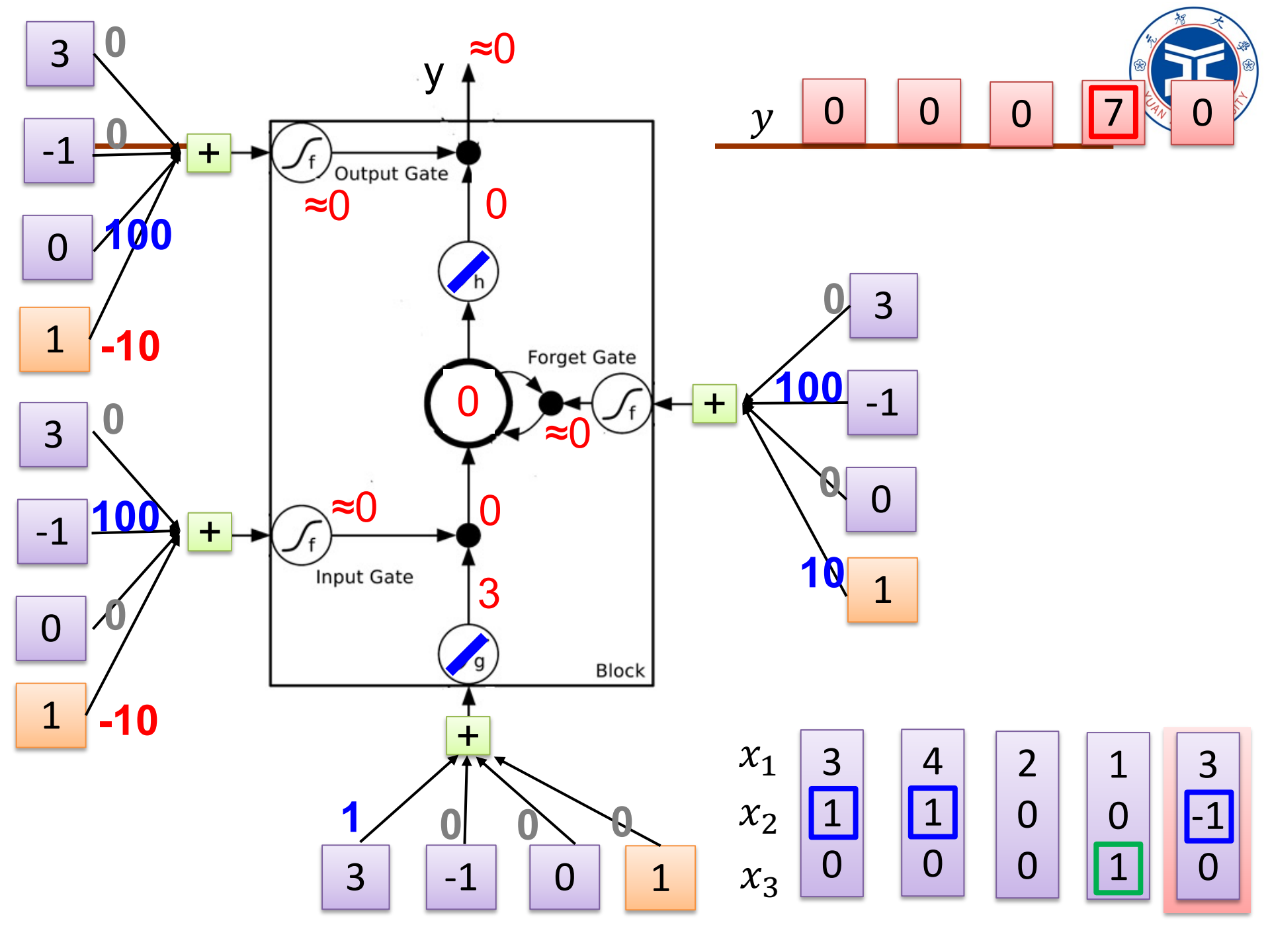

y
0 0 0 7 0
Output Gate
Forget Gate
Input Gate
Block
0
100
-10
1
x1
x2
x3
f
h
g
+


y
≈0
Output Gate
≈0
3
h
Forget Gate
3
≈1
Input Gate
≈1
3
3
g
Block
+
f
100
-10
100
-10
1
0
0
0
3
1
0
1
y 0 0 0 7 0
x_1 3 4 2 1 3
x_2 1 1 0 0 -1
x_3 0 0 0 1 0


y
≈0
Output Gate
≈0
7
Forget Gate
7
≈1
Input Gate
≈1
4
4
Block
4 0
1 0
0 100
1 -10
4 0
1 100
0 0
1 -10
0 4
100 1
0 0
10 1
1 4
0 1
0 0
0 1
y 0 0 0 7 0
x_1 3 4 2 1 3
x_2 1 1 0 0 -1
x_3 0 0 0 1 0

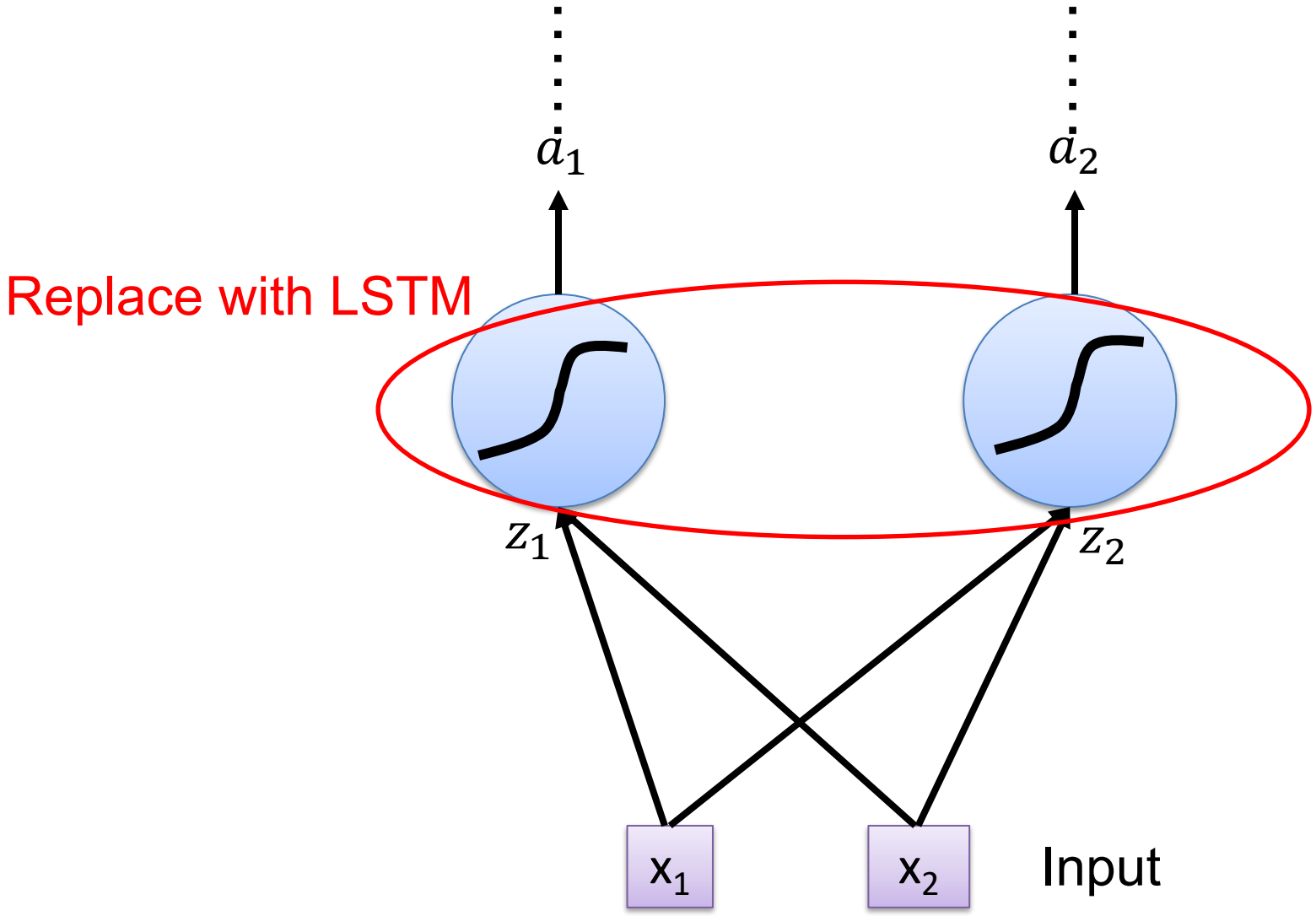

y
≈0
Output Gate
≈0
7
Forget Gate
7
≈1
Input Gate
≈0
0
2
Block
+
f
h
g
2
0
0
100
1
-10
0
100
0
-10
0
2
100
0
0
0
10
1
1
0
0
0
2
0
0
1
y 0 0 0 7 0
x_1 3 4 2 1 3
x_2 1 1 0 0 -1
x_3 0 0 0 1 0

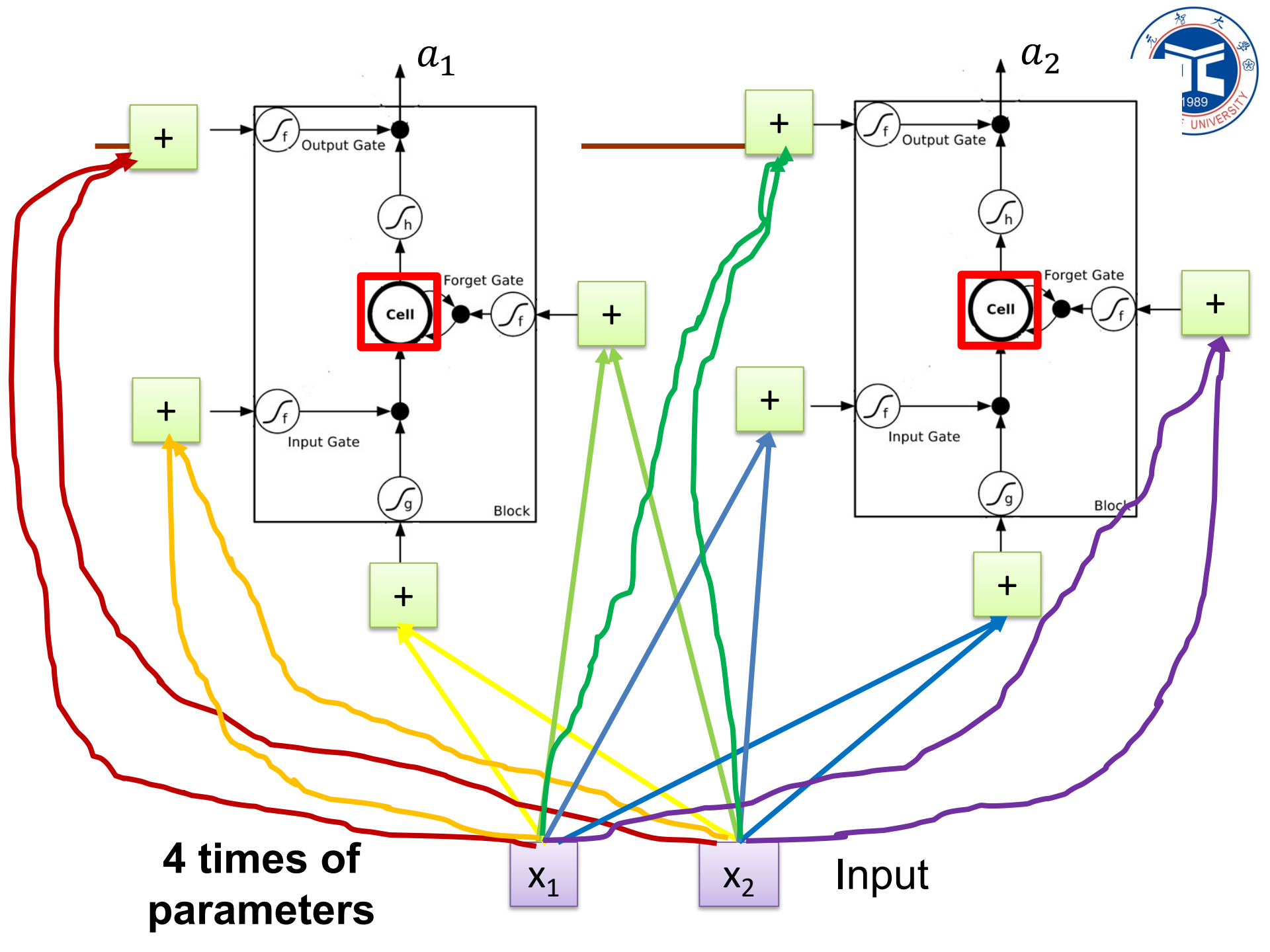

y
≈7
Output Gate
≈1
7
Forget Gate
≈1
Input Gate
≈0
0
1
Block
f
h
g
+
100
-10
10
1
0
y
0 0 0 7 0
x_1 3 4 2 1 3
x_2 1 1 0 0 -1
x_3 0 0 0 1 0


3
0
-1
0
0
100
1
-10
+
f
Output Gate
≈0
y
≈0
0
h
Forget Gate
0
≈0
f
+
0
3
100
-1
0
0
10
1
3
0
-1
100
0
0
1
-10
+
f
≈0
Input Gate
0
3
g
Block
+
1
0
0
0
3
-1
0
1
y
0
0
0
7
0
x_1
x_2
x_3
3
1
0
4
1
0
2
0
0
1
0
1
3
-1
0

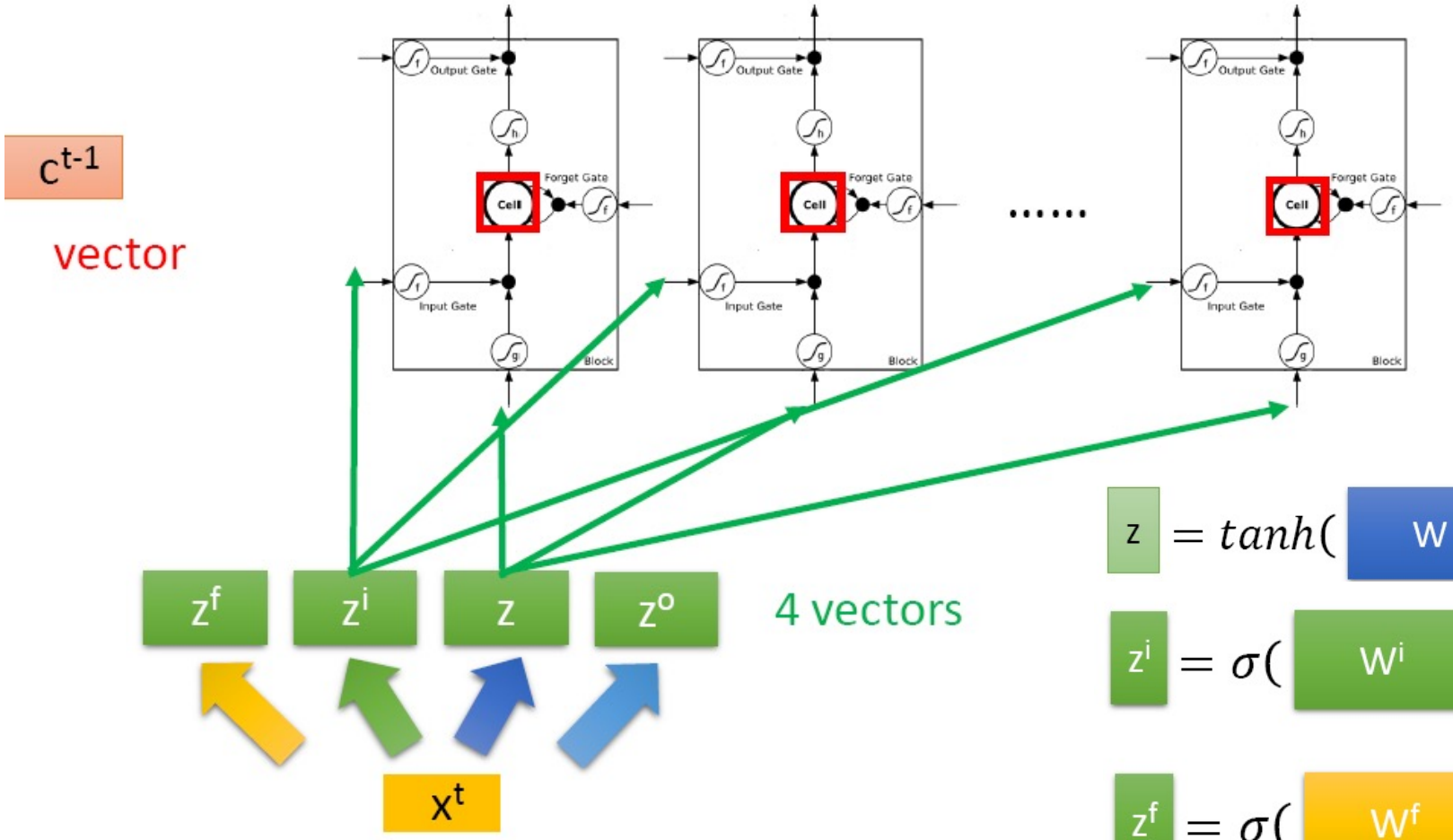
Original Network

- Simply replace the neurons with LSTM




a_1
a_2
Output Gate
Forget Gate
Input Gate
Cell
Block
x_1
x_2
Input
4 times of parameters

LSTM



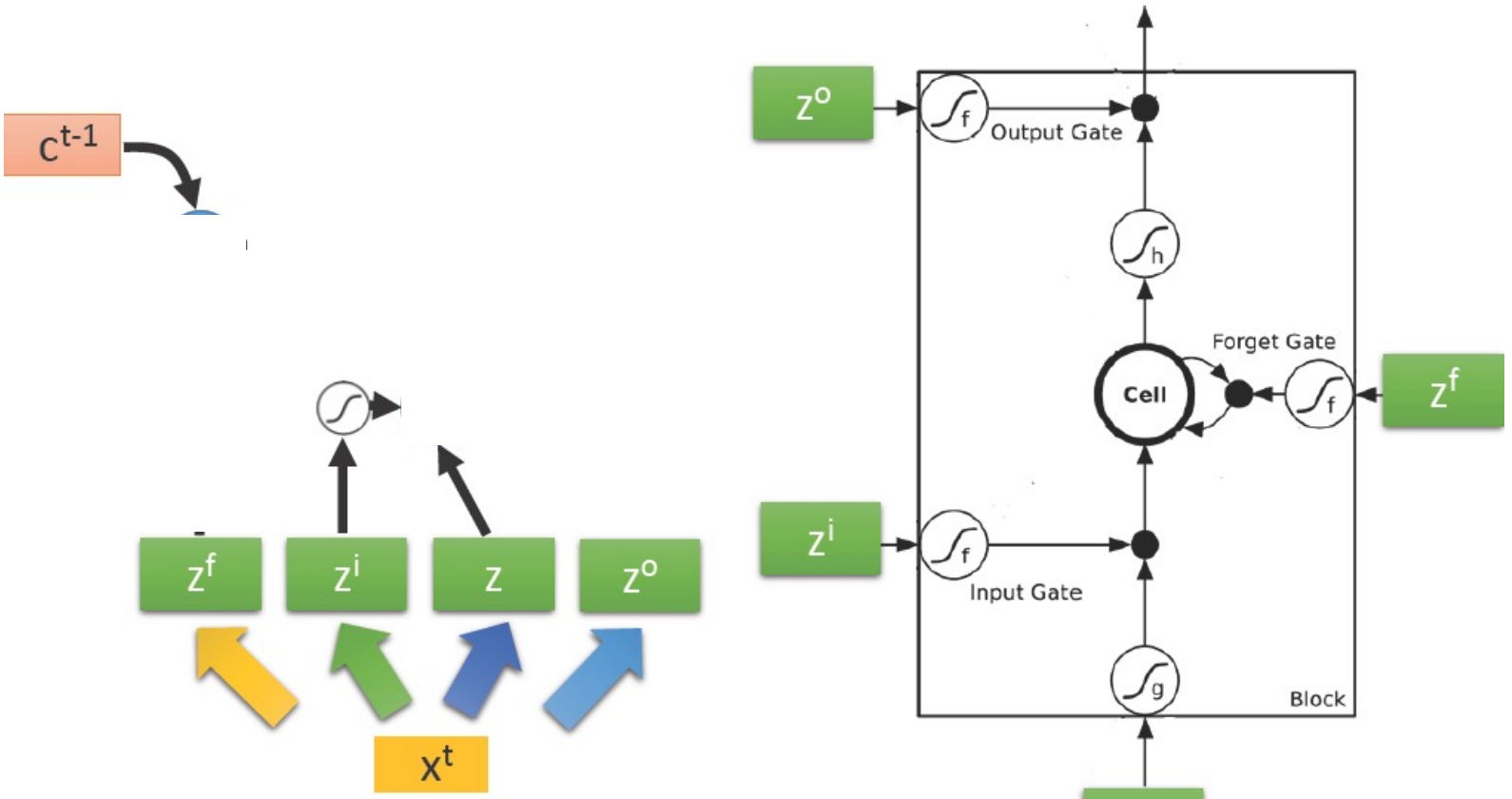
$$z = tanh(\ W\ x^t\ )$$

$$z^i = \sigma(\ W^i\ x^t\ )$$

$$z^f = \sigma(\ W^f\ x^t\ )$$

$$z^o = \sigma(\ W^o\ x^t\ )$$

LSTM



$$c^t = z^f \odot c^{t-1} + z^i \odot z$$

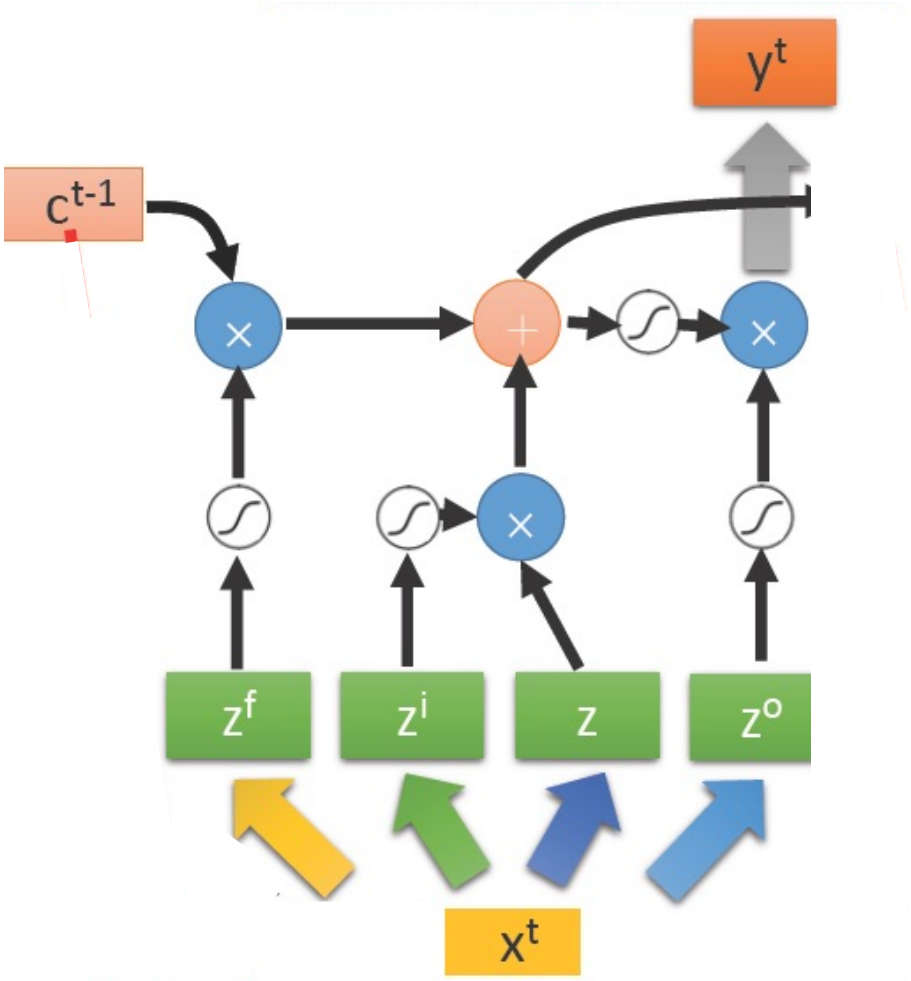
$$h^t = z^o \odot tanh(c^t)$$

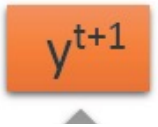
$$y^t = \sigma(W' h^t)$$

LSTM

Complex LSTM add last output as next input!

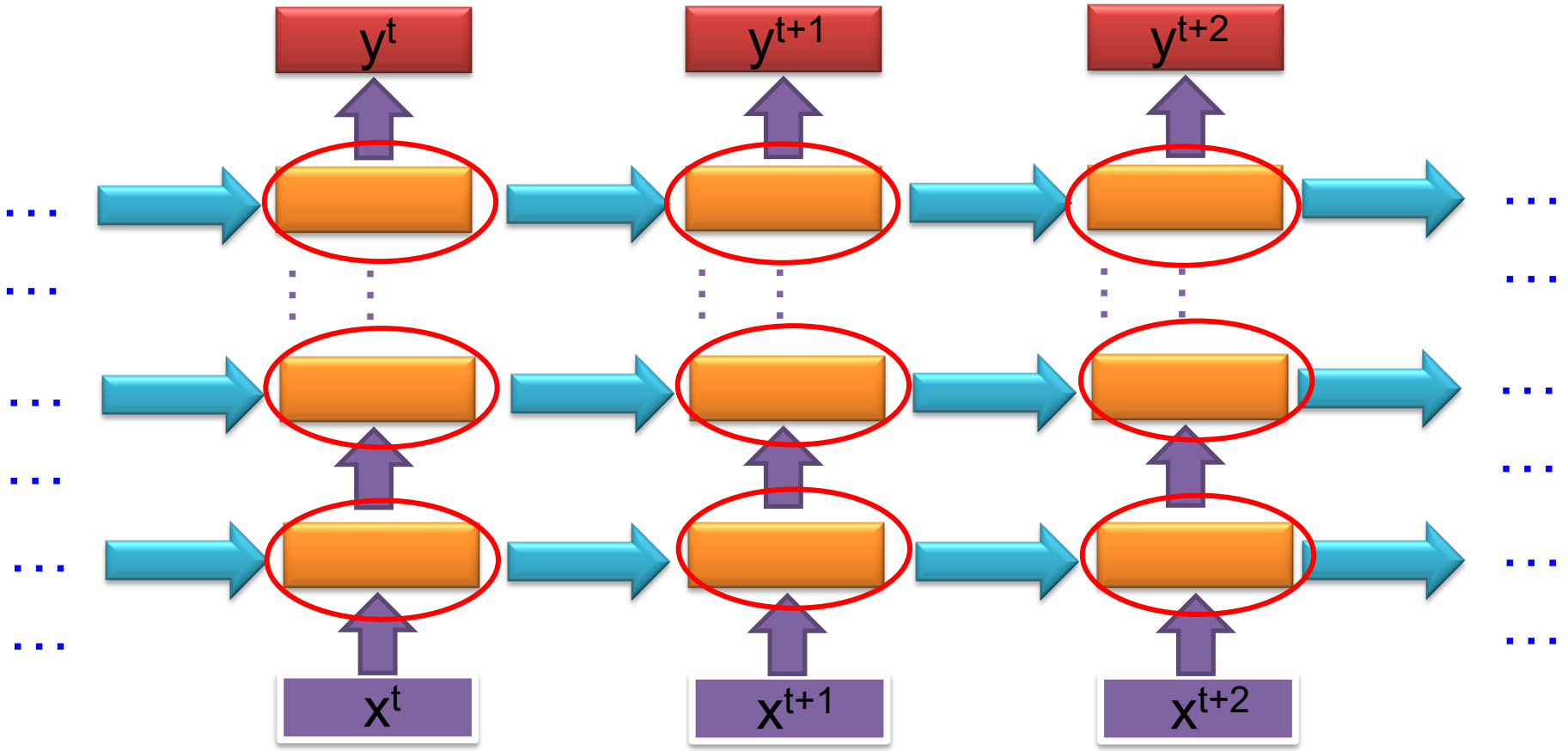
Peephole: last memory as next input!!



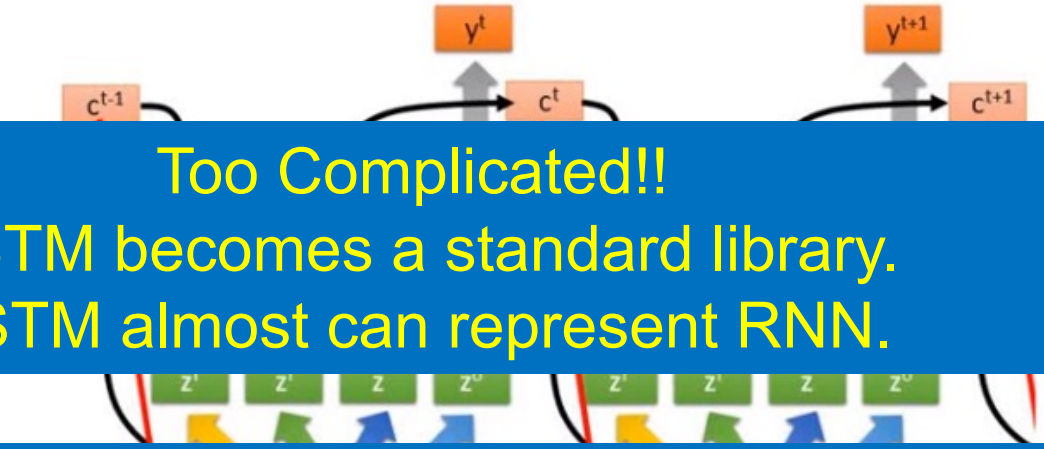


Deep RNN

- Replace neuron as LSTM

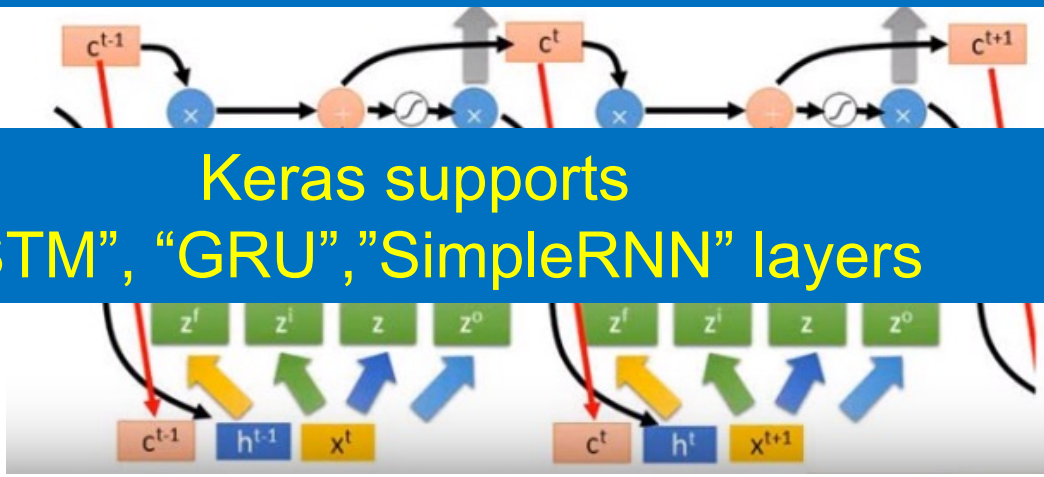


Multi-layer LSTM



Too Complicated!!
LSTM becomes a standard library.
LSTM almost can represent RNN.

Find "LSTM" in Keras.
You can import LSTM immediately.

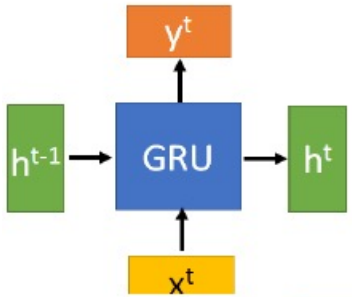


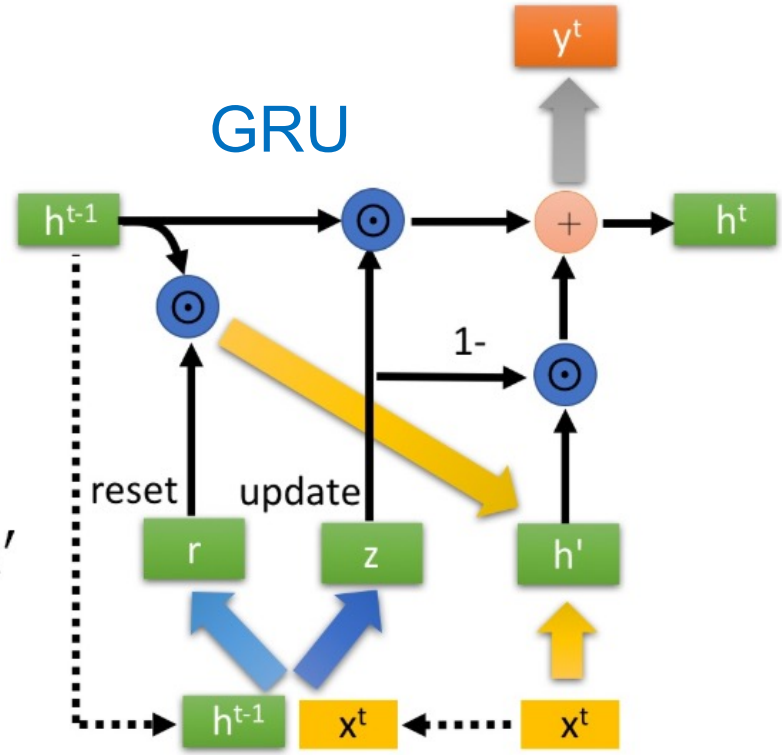
Keras supports
"LSTM", "GRU","SimpleRNN" layers

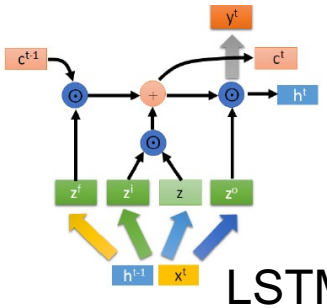
Keras is an open-source neural-network library written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, Theano, or PlaidML

Gate Recurrent Unit (GRU)

- Simplify LSTM
- Only two gate (reset, update gate)
  - parameters are fewer than LSTM
  - Prevent overfitting
- Input gate and forget gate are linked.
  - When there is new input, forget the older value.







LSTM

$$h^t = z \odot h^{t-1} + (1-z) \odot h'$$

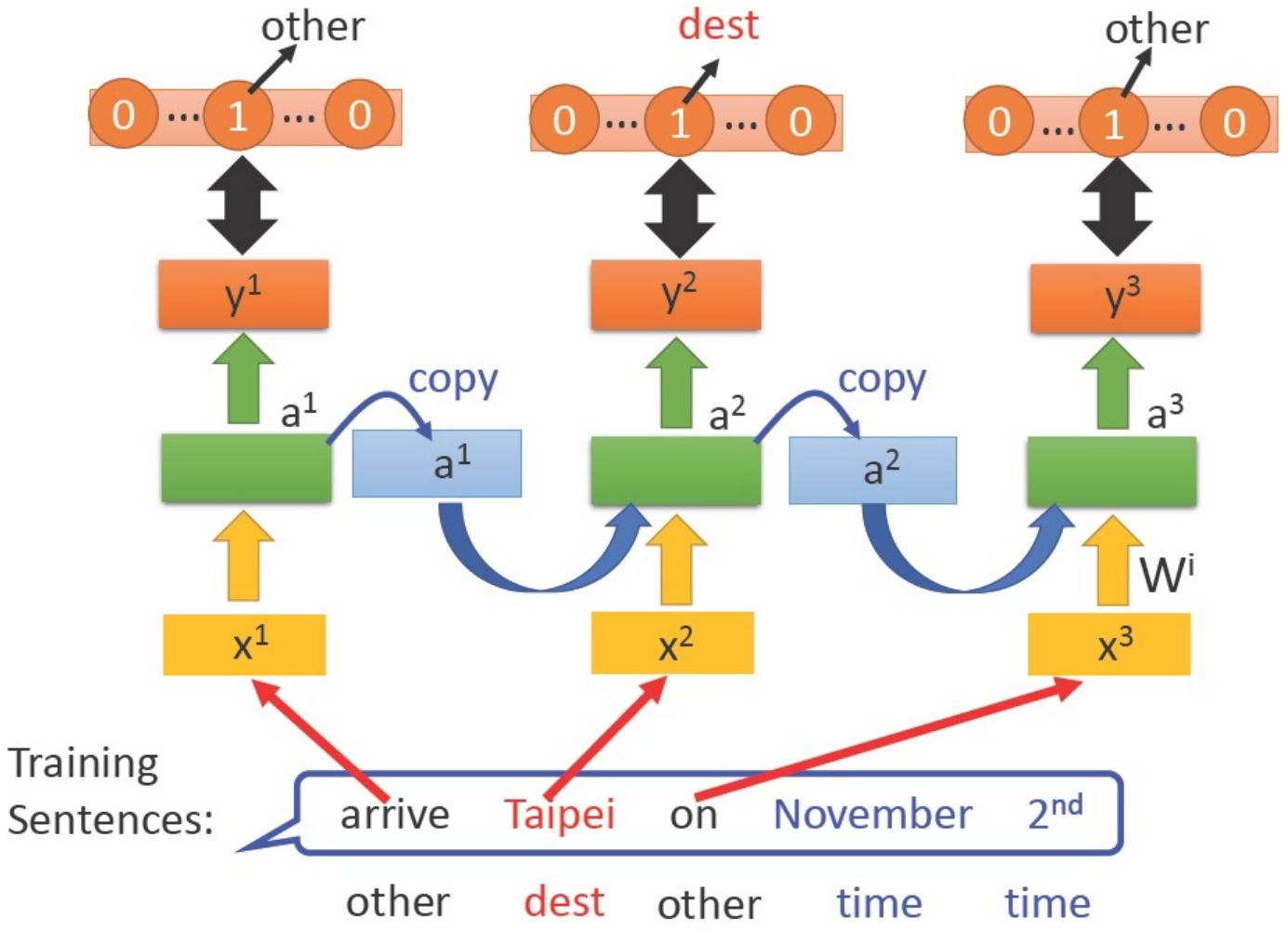
Similar

$$c^t = z^f \odot c^{t-1} + z^i \odot z$$

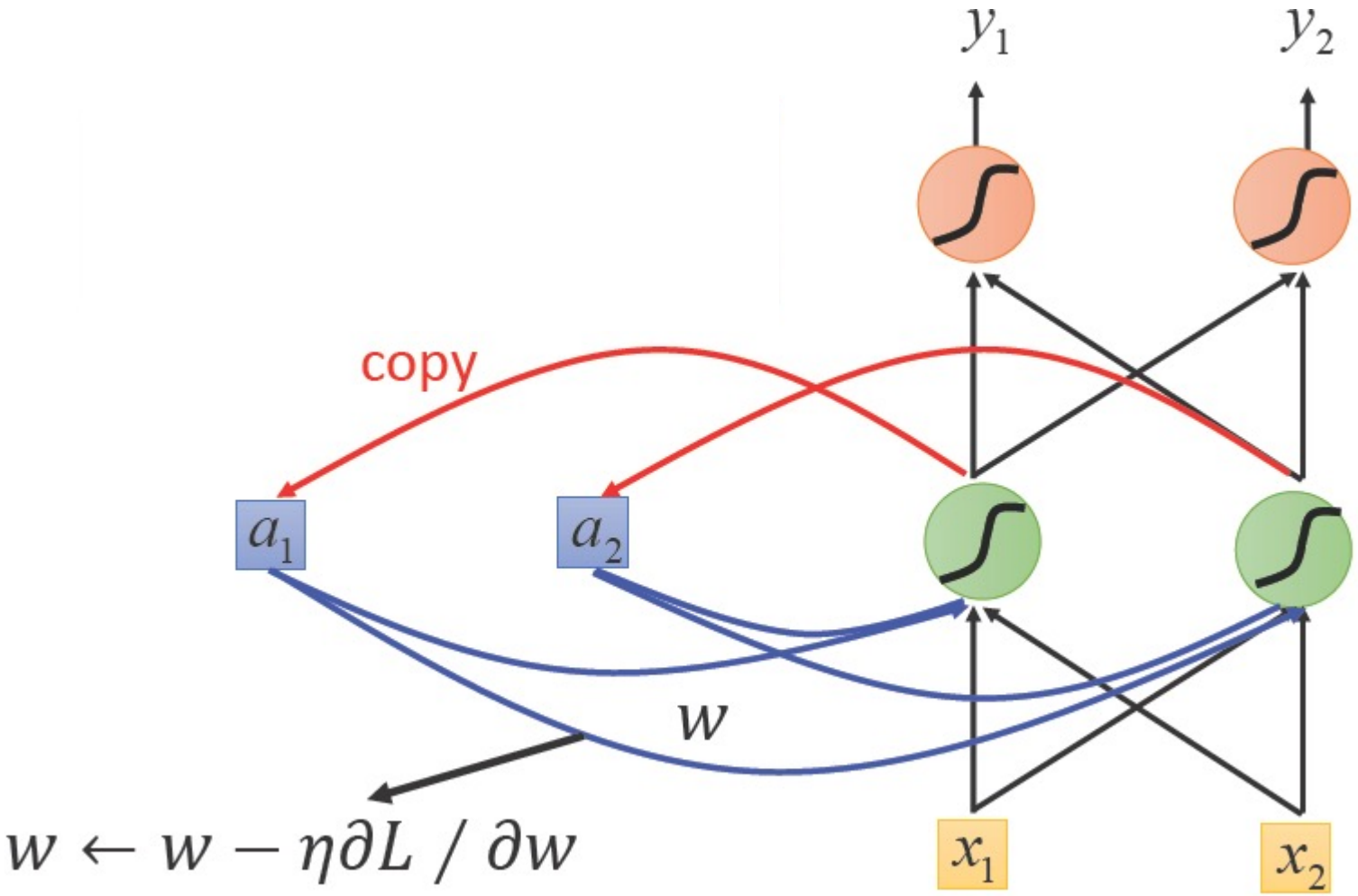
Learning in RNN

Learning Target

- Slot filling



Learning



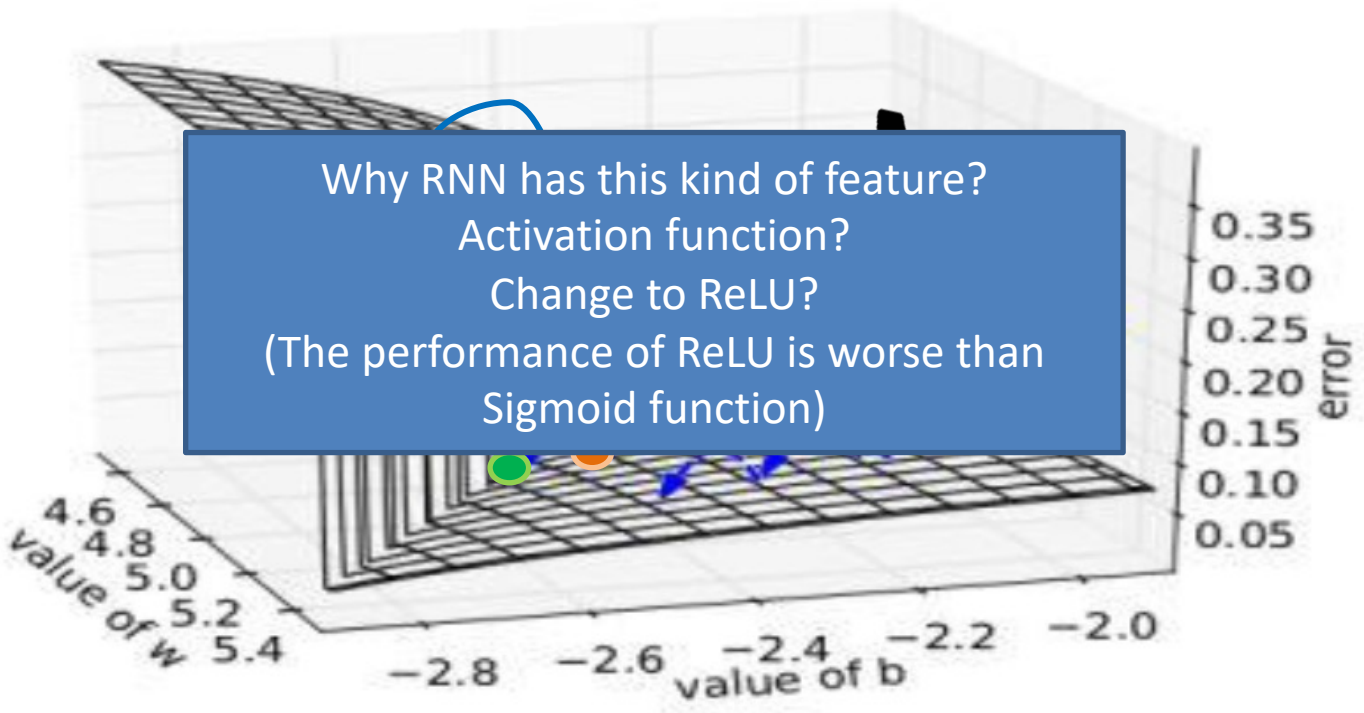
Unfortunately…..

- RNN-based network is not always easy to learn

The Error Surface is Rough

- Error surface: Total loss vs. Parameter changes
- Very flat or very steep on error surface in practice.

Clipping: When gradient is larger than threshold, gradient equals to current value and stop.



[Ravzan Pascanu, ICML'13]

Why?

$w = 1 \rightarrow y^{1000} = 1$

$w = 1.01 \rightarrow y^{1000} \approx 20000$

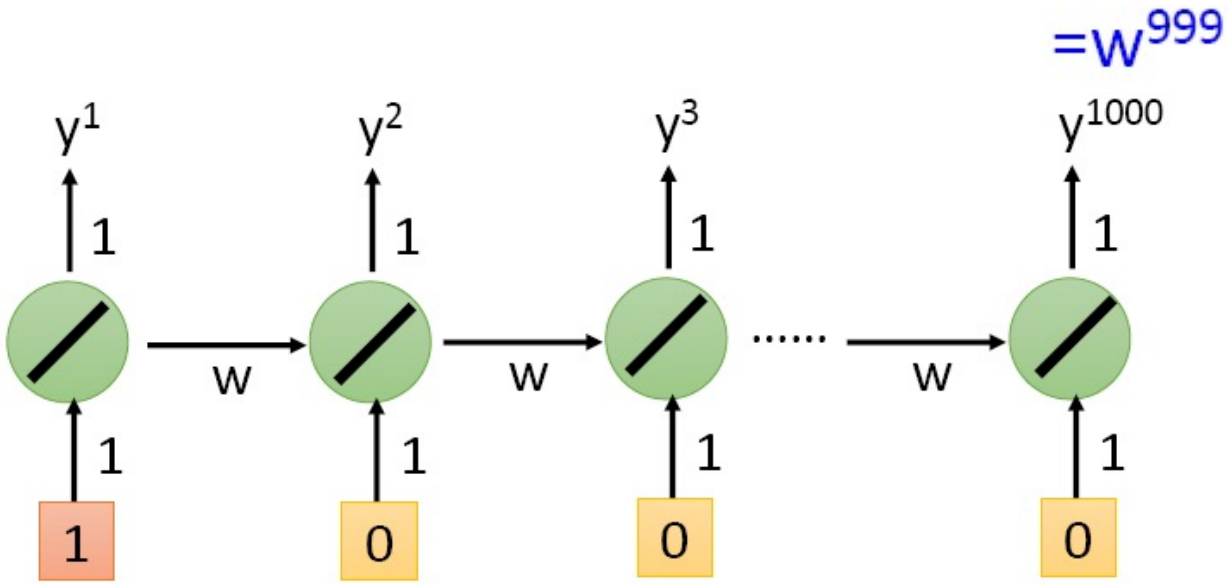
Large $\partial L/\partial w$ → Small Learning rate?

$w = 0.99 \rightarrow y^{1000} \approx 0$

$w = 0.01 \rightarrow y^{1000} \approx 0$

small $\partial L/\partial w$ → Large Learning rate?

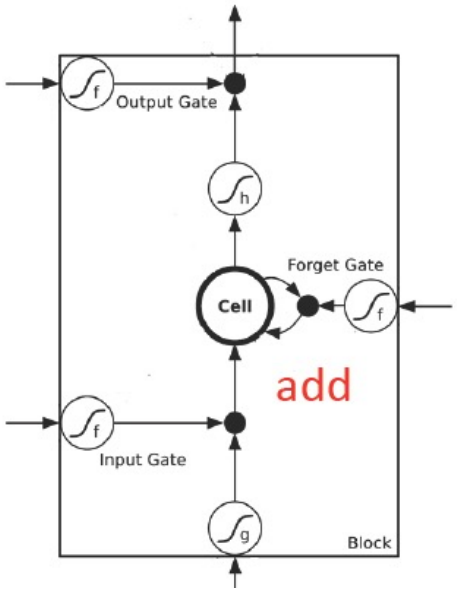
Toy Example



Helpful Techniques

- Long Short-term Memory (LSTM)
  - Can deal with gradient vanishing (not gradient explode)
  - Memory and input are added
  - The influence never disappears unless forget gate is closed.

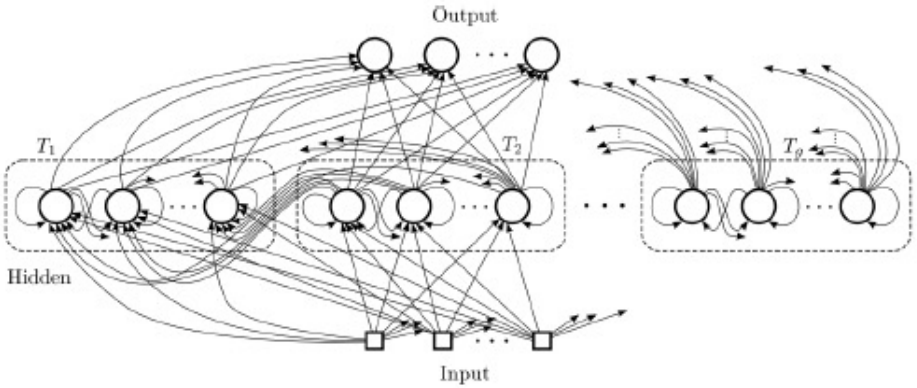
No Gradient vanishing
(If forget gate is opened.)



Gated Recurrent Unit(GRU): simpler than LSTM.
Training is more robust than LSTM.

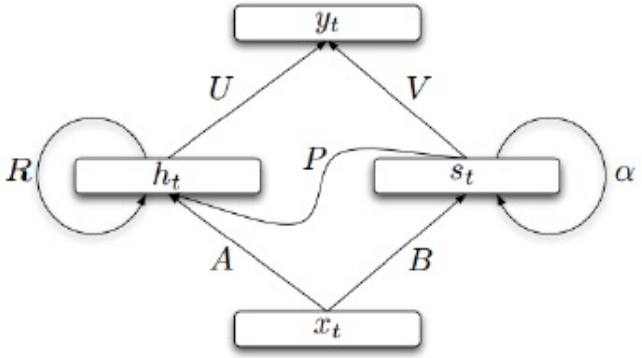
Helpful Techniques

Clockwise RNN



[Jan Koutnik, JMLR'14]

Structurally Constrained Recurrent Network (SCRN)



[Tomas Mikolov, ICLR'15]

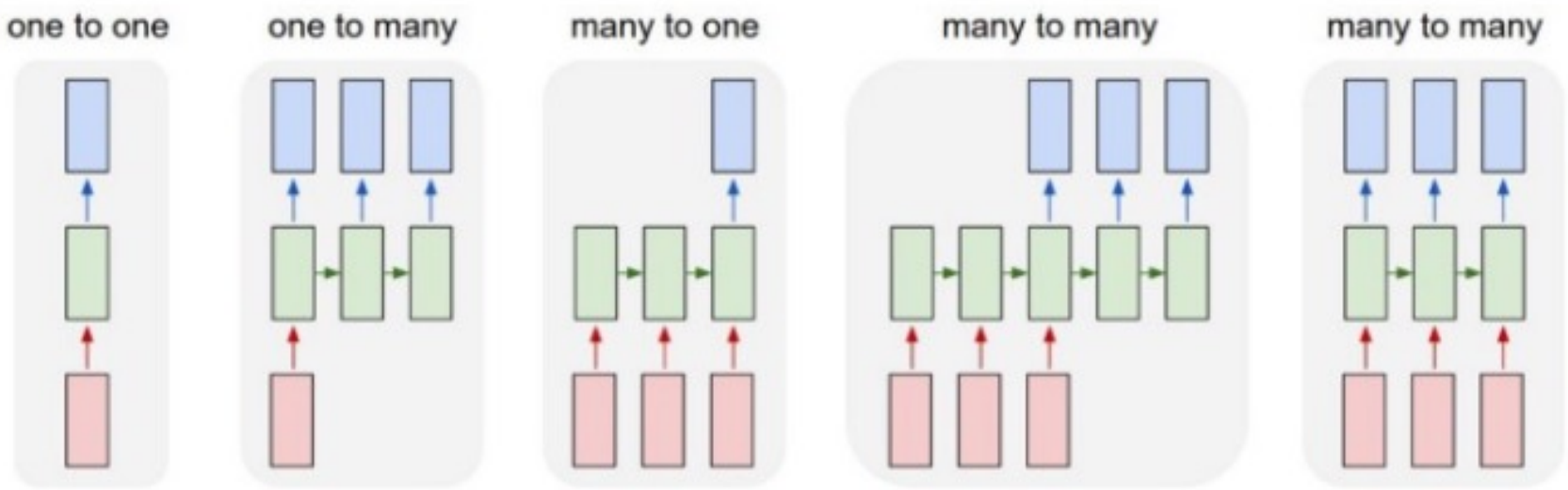
Vanilla RNN Initialized with Identity matrix + ReLU activation function [Quoc V. Le, arXiv'15]

- Outperform or be comparable with LSTM in 4 different tasks

More Applications of RNN

Multiple Models

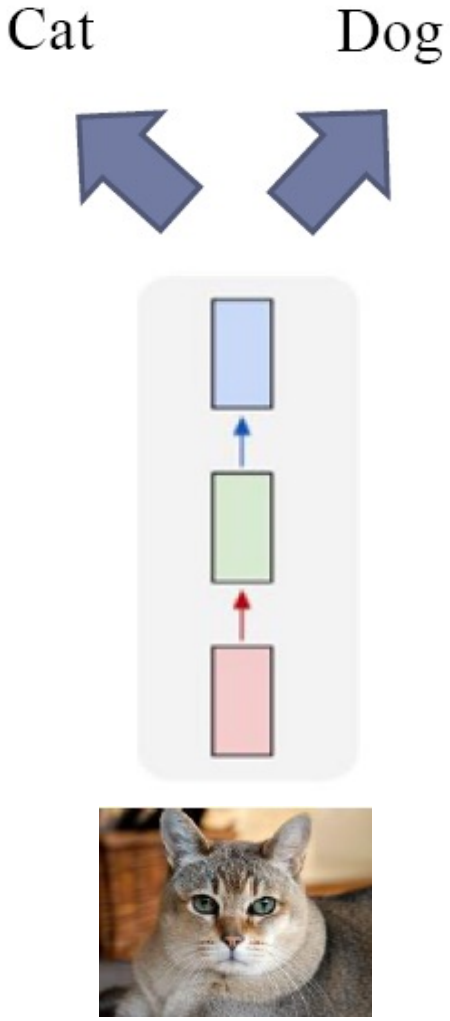
- The number of input and output of RNN can be different.



EX: POS Tagging

One to One

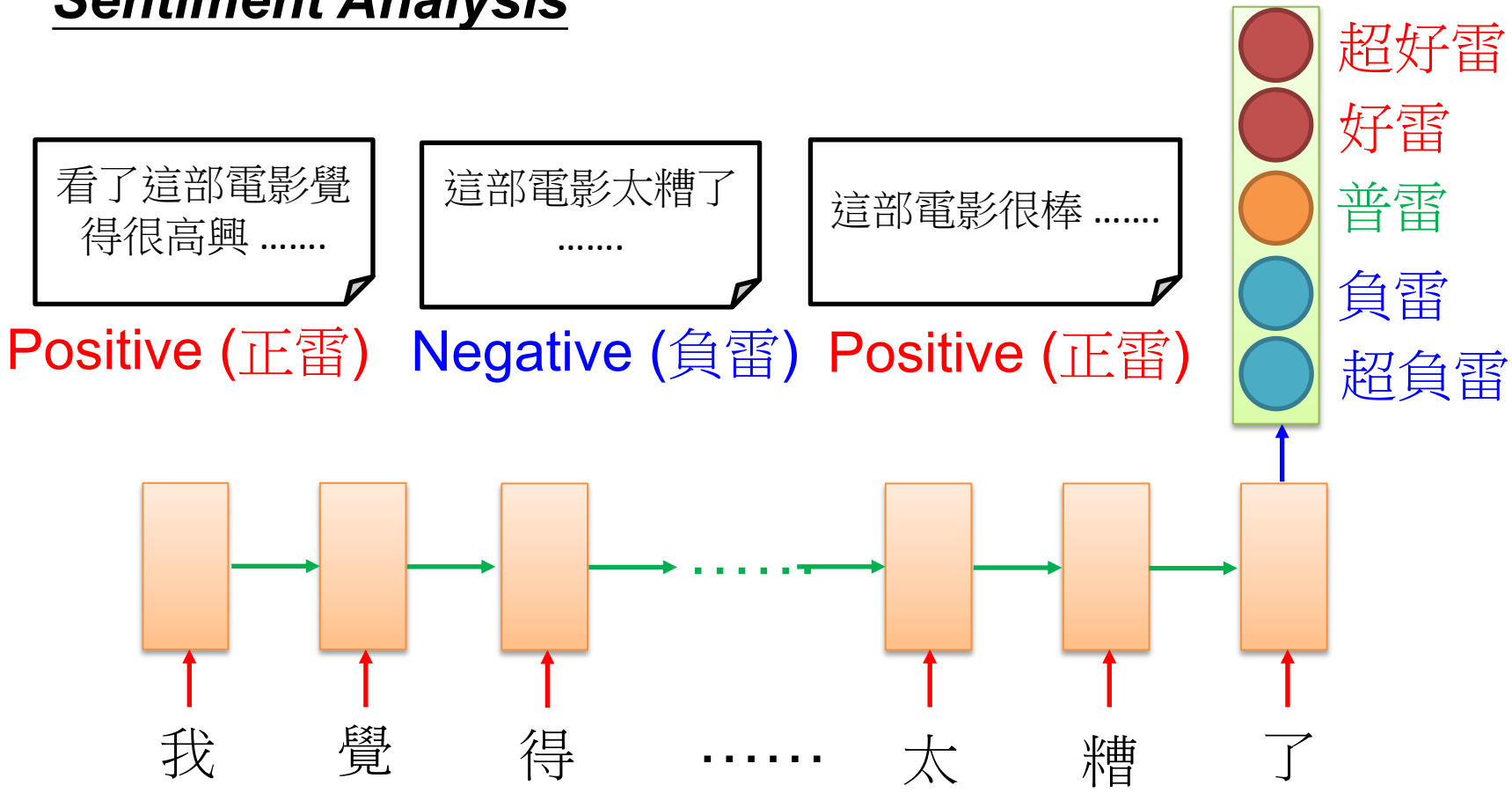
- Classification task
- DNN



Many to One

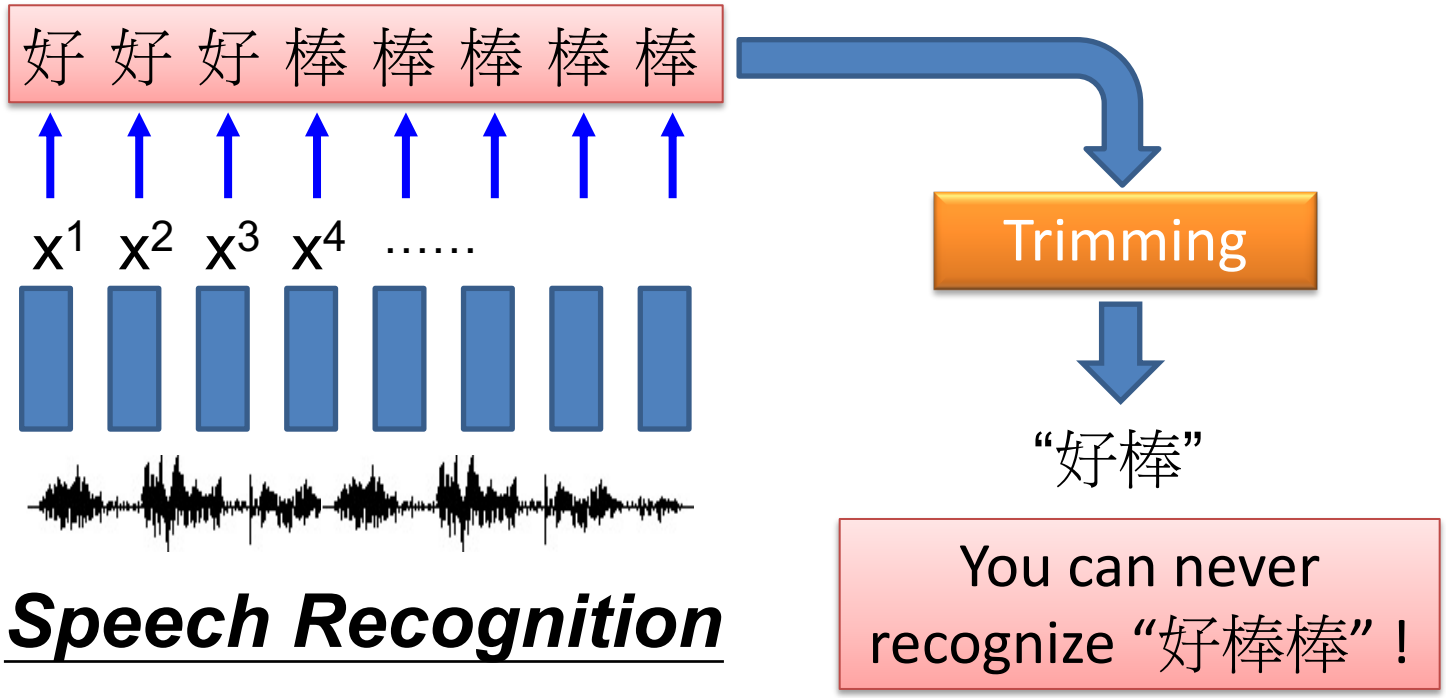
- Input is a vector sequence, but output is only one vector

Sentiment Analysis



Many to Many (Output is shorter)

- Both input and output are vector sequences, ***but the output is shorter.***



Speech Recognition

Many to Many (Output is shorter)

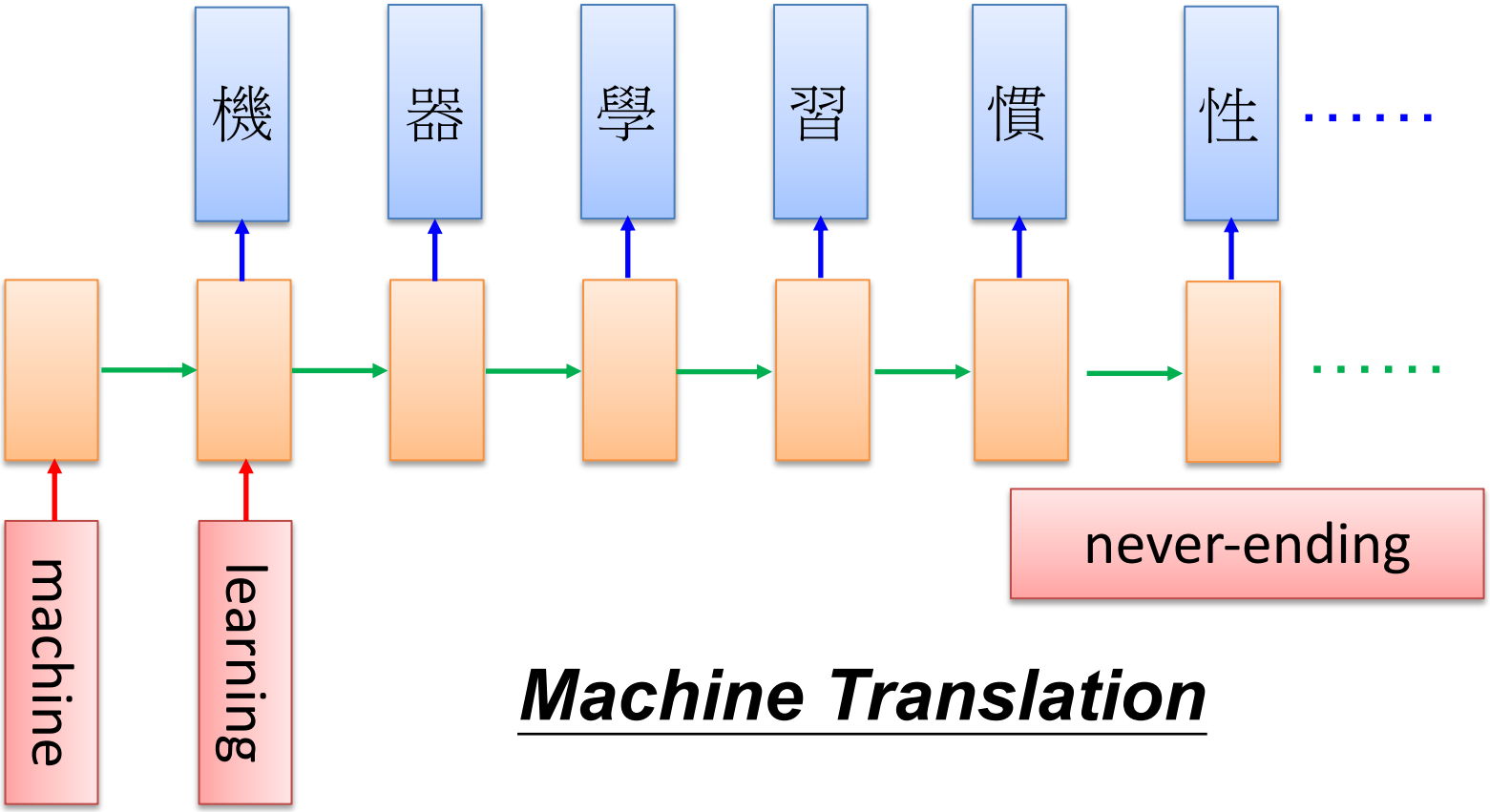
- Both input and output are vector sequences, ***but the output is shorter.***
- Connectionist Temporal Classification (CTC)
  - Add an extra symbol "φ" (同上)

好 φ φ 棒 φ φ φ φ → "好棒"

好 φ φ 棒 φ 棒 φ φ → "好棒棒"

Many to Many (No Limitation)

- Both input and output are vector sequences ***with different lengths***. → ***Sequence to sequence learning***



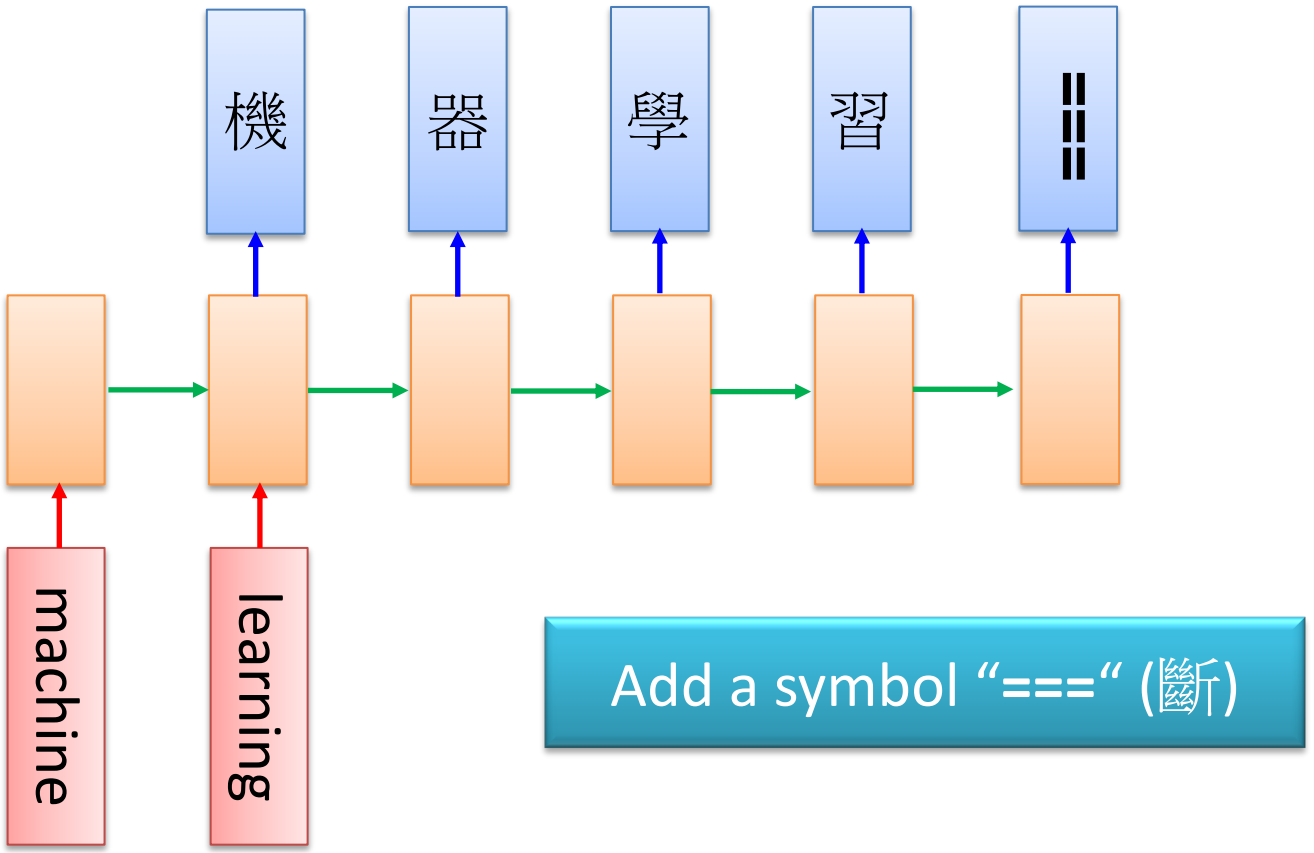
Many to Many (No Limitation)

- 推文接龍
  - Ref: http://pttpedia.pixnet.net/blog/post/168133002-%E6%8E%A5%E9%BE%8D%E6%8E%A8%E6%96%87

推xxx: ptt萬歲
推dd: 歲平安
噓dddf: 全
推zzzzzzzzzz: 家就是你家
⋮ ⋮
推tlkagk: ==========斷==========

Many to Many (No Limitation)

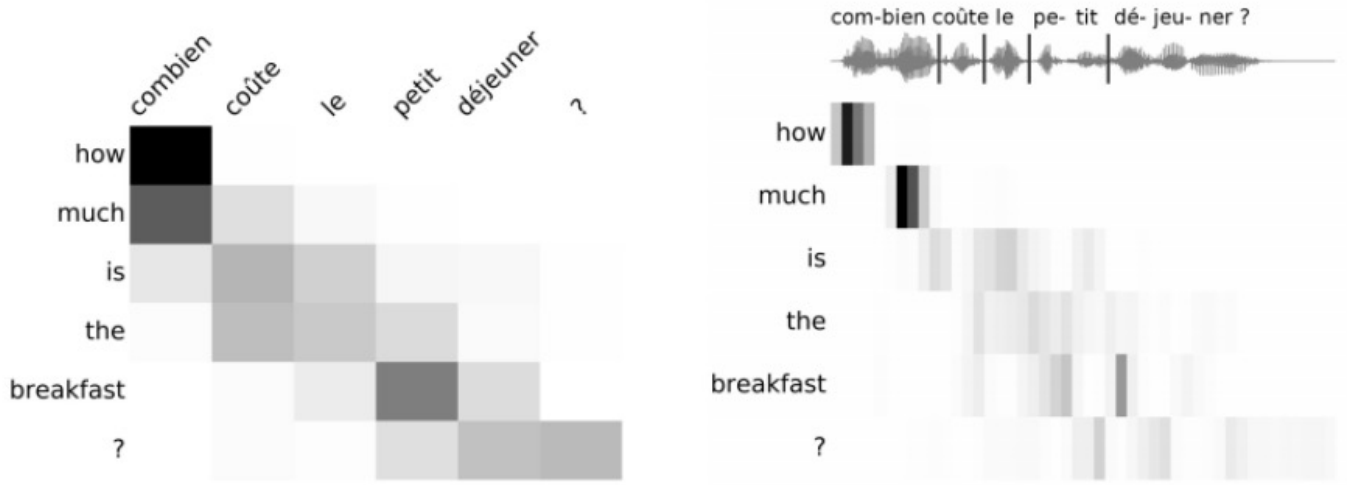
- Both input and output are vector sequences ***with different lengths***. → ***Sequence to sequence learning***



Many to Many (No Limitation)

- Both input and output are both sequences with different lengths. ->Sequence to sequence learning
- EX: Machine Translation (English translates to Chinese).

Is there possible input sounds of English and output the text of Chinese without speech recognition?



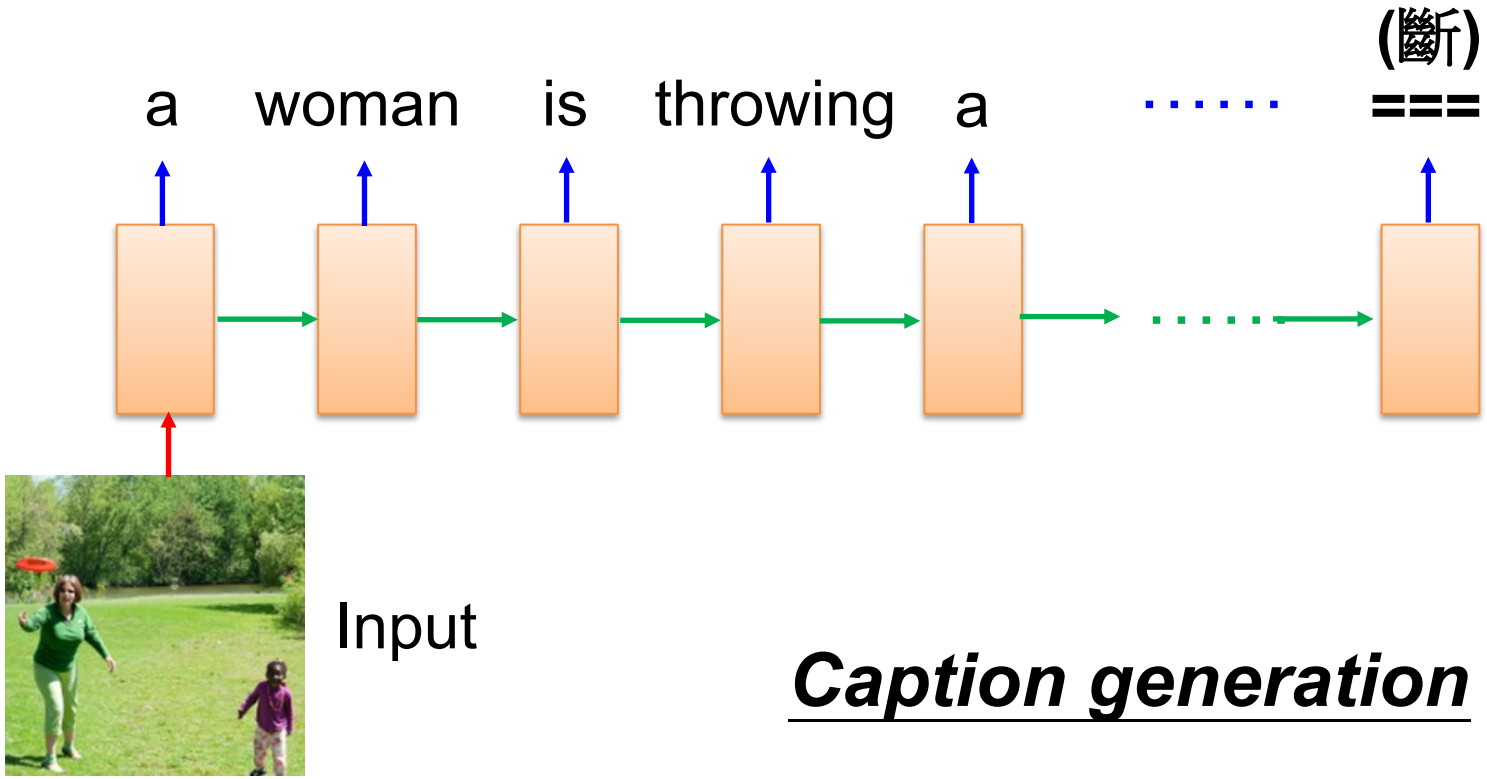
(a) Machine translation alignment (b) Speech translation alignment

Figure 1: Alignments performed by the attention model during training

https://arxiv.org/pdf/1612.01744v1.pdf

One to Many

- Input is one vector, but output is a vector sequence



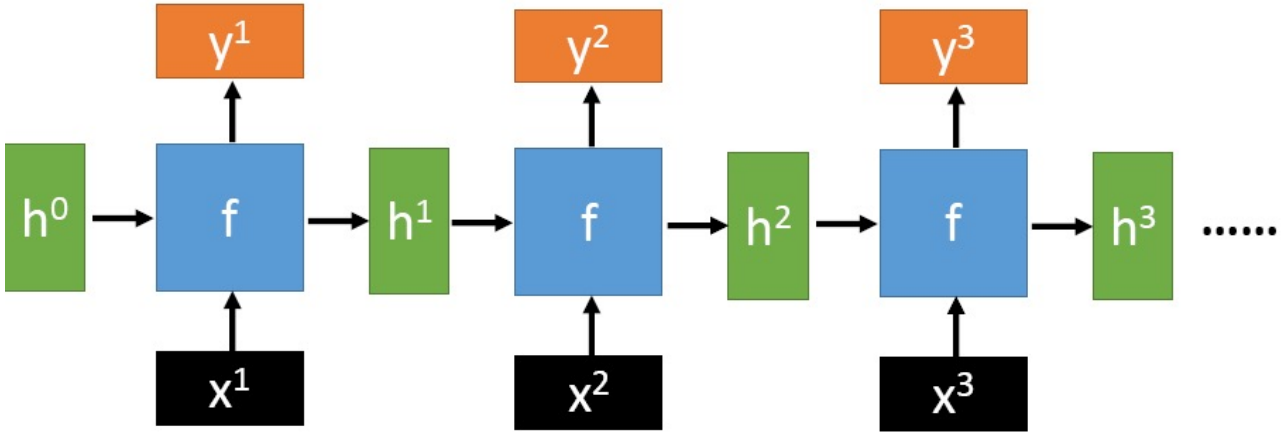
Caption generation

RNN for Anomaly Detection

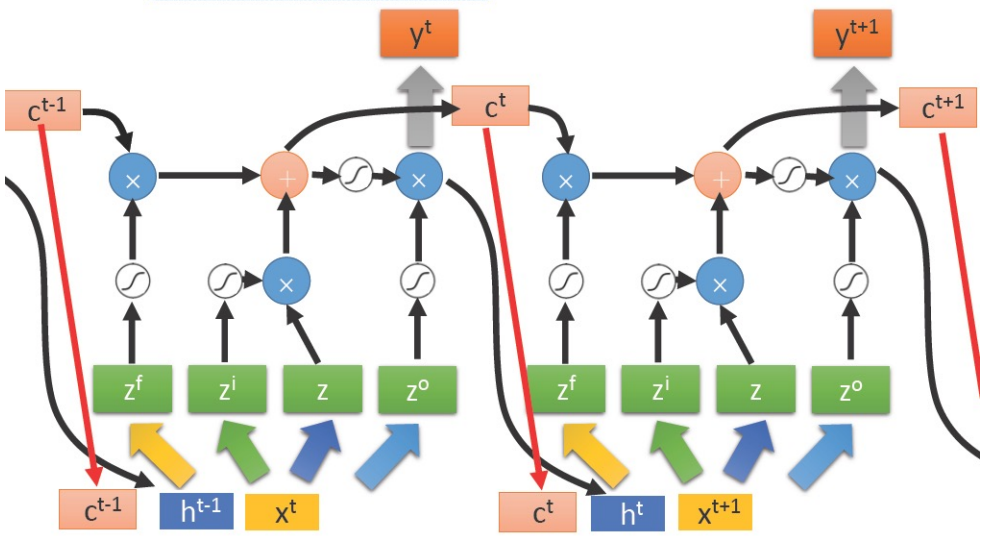
Summary

Summary

- Recurrent Neural Network



- LSTM



Recommended Reading List

- The Unreasonable Effectiveness of Recurrent Neural Networks
  - http://karpathy.github.io/2015/05/21/rnn-effectiveness/
- Understanding LSTM Networks
  - http://colah.github.io/posts/2015-08-Understanding-LSTMs/